UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/087,347	03/01/2002	Mark E. Rorvig	4380.001300-	3070

7590 01/13/2005

Williams Morgan & Amerson
7676 Hillmont
Suite 250
Houston, TX 77040

EXAMINER	
SETH, MANAV	
ART UNIT	**PAPER NUMBER**
2625	

DATE MAILED: 01/13/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/087,347	RORVIG ET AL.	
	Examiner	Art Unit	
	Manav Seth	2625	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *01 March 2002*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-52* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-52* is/are rejected.

7)☒ Claim(s) *15,41* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *12/02/2003*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Claim Objections

1. Claims 15 and 41 are objected to because of the following informalities:

Claim 15 recites the limitations "constructing a second histogram for each of the selected features using the second measurement information; determining a **first** (assumed for examining purpose: **second**) area encompassed by each of the **first** (assumed for examining purpose: **second**) histograms; encoding the **first** (assumed for examining purpose: **second**) areas of the **first** (assumed for examining purpose: **second**) histograms in metadata elements of a **first** (assumed for examining purpose: **second**) hypertext markup language (HTML) document; and".

Claim 41 "means for constructing a second histogram for each of the selected features using the second measurement information; means for determining a **first** (assumed for examining purpose: **second**) area encompassed by each of the **first** (assumed for examining purpose: **second**) histograms; means for encoding the **first** (assumed for examining purpose: **second**) areas of the **first** (assumed for examining purpose: **second**) histograms in metadata elements of a **first** (assumed for examining purpose: **second**) hypertext markup language (HTML) document; and".

Applicant in claims 15 and 41 recites the measurement of selected features of first object and second object. In these claims, first area under first histograms are

associated with first object and further linked to first HTML document and again first area under first histograms are associated to second object and further linked to first HTML document. These limitations are inconsistent with the limitations:

1. Claim 15 - "constructing a **second** histogram for each of the selected features using the second measurement information".

2. Claim 41 – "means for constructing a **second** histogram for each of the selected features using the second measurement information".

Therefore, it is assumed by the examiner for the examining purposes that a second area under second histograms is calculated for second object and are further linked to second HTML document.

Appropriate correction is required.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1, 2, 27, 28, 40 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abdel-Mottaleb et al., U.S. Patent No. 5,915,038 in view of Tsymbalenko et al., Jan. 9, 2001, "Using HTML Metadata to find relevant images on the world wide web".

Claim 1 recites "retrieving a first object, wherein the first object comprises a first measurement information encoded in metadata elements of a first hypertext mark up language (HTML) document". Abdel-Mattaleb discloses computing an index key (metadata) for a query (first) image where as the query (first) image is an image identified (retrieved) by the user for use in retrieving other similar images (column 3, lines 40-43; column 4, lines 38-43). Abdel-Mettaleb further discloses the extraction of index keys is the extraction of metadata, where extracting metadata is the measurement of the features of the still images (column 7, lines 17-23).

Claim 1 further recites "comparing the first object with a second object, wherein the second object comprises a second measurement information encoded in metadata elements of a second hypertext markup language (HTML) document". Abdel-Mettaleb discloses the comparison of the index key (metadata information) of the query (first) image with each index key of the corresponding image (second) being searched (column 3, lines 45-50).

Claim 1 recites "retrieving 'the second object in response to the difference between the first measurement information of the first HTML document and the second measurement information of the second HTML document being less than or equal to a

threshold difference value". Abdel-Mettaleb discloses the selection of the second image with respect to the threshold difference value (column 3, lines 50-60).

Abdel-Mettaleb further discloses that the still images may be stored locally or remotely, such as a remote Web site on the Internet and archived using the index key extraction and archival process (column 8, lines 13-16). Abdel-Mettaleb does not explicitly teach the encoding of metadata information in metadata elements of a HTML document.

However, Tsymbalenko discloses the image retrieval system where the images are accessed through HTML documents and that the bulk of the content of HTML documents is textual whereas the textual content and the structure of HTML documents are considered to be "metadata" describing the images and this metadata is used to determine which images may be relevant to a query (page 3, paragraph 2). Tsymbalenko further discloses the search strategy where the result showed a HTML document with links to other HTML documents that best matched the query (page 3, paragraph 5) and every image is linked to each HTML document (page 3, para. 6, lines 2-3).

Therefore, it would had been obvious to one having ordinary skill in the art at the time of the invention was made to include the teachings by Tsymbalenko in the invention of Abdel-Mettaleb. One would have been motivated to use the concept of image metadata encoding in the HTML documents by Tsymbalenko in the invention of Abdel-Mettaleb because both the references are directed to image search and retrieval. Abdel-Mettaleb discloses that the still images may be stored locally or remotely, such as

a remote web site on the internet and Tsymbalenko further provides the image search and retrieval using the HTML documents where each image index key (metadata) extracted from Abdel-Mettaleb's invention can be encoded in HTML document as a textual information and the advantage of using metadata textual information would lead in saving memory and bandwidth on the image retrieval system.

Claim 2 has been analyzed and rejected as per claim 1.

Claim 27 additionally recites the system that retrieves images by content measure metadata encoding. Abdel-Mettaleb discloses the use of computer system for his invention (column 6, lines 24-37). All other limitations in claim 27 had been analyzed and rejected as per claim 1.

Claim 28 has been analyzed and rejected as per claims 1 and 27.

Claim 40 has been analyzed and rejected as per claim 27.

4. Claims 3, 9-12, 14, 29, 35-38 and 40 are rejected under 35 U.S.C. 103(a) as being unpatentable over Eakins et al., Jan. 1999, "Content-based image retrieval: a report to the JISC technology applications programme", in view of Opittek et al., U.S. Patent No. 3,979,555, and further in view of Tsymbalenko et al., Jan. 9, 2001, "Using HTML Metadata to find relevant images on the world wide web".

Claim 3 recites "A method of encoding images by content measure metadata encoding, comprising: measuring selected features of an object to form measurement information; constructing a histogram for each of the selected features using the measurement information". Eakins discloses content-based image retrieval (CBIR) methods (page 18) and systems (page 23) where stored images are retrieved from a collection by comparing features automatically extracted from the **images** themselves where the commonest features used are mathematical measures of **color, texture or shape** (page 18, para. 3). Eakins further discloses measuring the proportion of pixels of **each** color within the image and a color histogram is computed to show this measurement (page 18, para. 4, lines 2-4).

Claim 3 further recites "determining an area encompassed by each of the histograms". Eakins does not teach of calculating an area encompassed by histogram for each of the selected features. However, Opittek discloses **calculating a total area under histogram** to compute the total distribution of gray scale intensity levels in the image (column 3, lines 13-20). Opittek discloses a histogram in figure 1-4 where x-axis represents internal division of the selected feature (intensity of a color) and y-axis represents a frequency of occurrence of intensity level.

Therefore, it would had been obvious to one having ordinary skill in the art at the time of the invention was made to use the concept of calculating area under the histogram by Opittek in the invention of Eakins. One would have been motivated to use the concept of calculating area under the histogram by Opittek in the invention of Eakins

because Eakins computes a color histogram for each color within the image and Opittek's further shows the histogram in standard x-y coordinate format with x-axis representing internal division of the selected feature (intensity of a color) and y-axis representing a frequency of occurrence of intensity level in figure 1 and then **calculates a total area under this histogram** to compute the total distribution of color intensity levels in the image and calculating area will provide a mathematical value of the distribution of the color intensity levels.

Claim 3 further recites "encoding areas of the histograms in metadata elements of a hypertext markup language (HTML) document". Eakins discloses the encoding of parameters such as the color space on which the histogram is based in metadata elements of a XML (Extensible Markup Language) document. Eakins does not teach encoding in metadata elements of a HTML document. Encoding in metadata of HTML is well known in the art and this is supported by Tsymbalenko et al., which further discloses that each image is associated to each HTML document as explained n claim 1.

Claim 9 recites "A method of claim 4, wherein measuring selected features further comprises measuring a geometric feature of the object". The features such as shape, size, distance, rotation angle, etc., are standard types of geometric features depending on the object type Therefore claim 9 has been analyzed and rejected as per claim 3.

Claim 10 and 11 had been analyzed and rejected as per claim 9.

Claim 12 has been analyzed and rejected as per claim 3.

Claim 14 has been analyzed and rejected as per claim 3 and in further in view of claim 1.

Claim 29 recites a system that performs the method as recited in claim 3. Eakins further discloses many content-based image retrieval systems and these systems use the same method as recited in claim 3 (page 23; page 18, para. 3, lines 3-4). Therefore, claim 29 has been analyzed and rejected as per claim 3.

Claim 35 has been analyzed and rejected as per claim 29 and further in view of claims 3 and 9.

Claims 36 and 37 had been analyzed and rejected as per claim 35.

Claim 38 has been analyzed and rejected as per claim 29 and further in view of claim 3 and 12

Claim 40 has been analyzed and rejected as per claim 29 and in further view of claims 3,14 and 1.

5. Claims 4-8 and 30-34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Eakins et al., Jan. 1999, "Content-based image retrieval: a report to the JISC technology applications programme", in view of Opittek et al., U.S. Patent No. 3,979,555, and further in view of Tsymbalenko et al., Jan. 9, 2001, "Using HTML Metadata to find relevant images on the world wide web", and further in view of Baxes, 1994, Book Publication, "Digital image processing: principles and applications".

Claim 4 recites "A method of claim 3, wherein measuring selected features further comprises measuring an intensity of a preselected color of the object". As explained in rejection of claim 3, the combined invention of Eakins and Opittek determines intensity levels of each color based on the histograms. Therefore claim 4 has been analyzed and rejected as per claim 3.

It is well known in the art that each color can be red, green or blue as in RGB space and each color can also be a gray color in a gray-scale image. This well known art is further support by Baxes, who in his book printed in 1994, discloses that three histograms can be generated in an RGB space for each color component (page 63, para 3; page 64, para 1 and figures 3.20(a), 3.20(b) and 3.20(c). Therefore, it would had been obvious to one having ordinary skill in the art at the time of the invention was made to use the concept of constructing histograms for a preselected color as disclosed by Baxes. One would have been motivated to use the concept of Baxes because each histogram can help to determine the brightness distributions, contrast, and dynamic

ranges of the individual components where brightness is the measured intensity of the pixels in the image (page 3).

Claims 5, 6, 7, and 8 had been analyzed and rejected as per claim 4.

Claims 30, 31, 32, 33, 34 had been analyzed and rejected as per claim 29 and in further view of claims 4, 5, 6, 7 and 8.

6. Claims 13 and 39 are rejected under 35 U.S.C. 103(a) as being unpatentable over Eakins et al., Jan. 1999, "Content-based image retrieval: a report to the JISC technology applications programme", in view of Opittek et al., U.S. Patent No. 3,979,555, and further in view of Tsymbalenko et al., Jan. 9, 2001, "Using HTML Metadata to find relevant images on the world wide web" and further in view of Rorvig, "An experimental approach for the content-based image analysis: an open source agenda for research".

Claim 13 recites "a method of claim 3, further comprising converting the area under the histogram to a Lorenz Information Measure (LIM)". Eakins, Opittek and Tsymbalenko do not teach converting the area under the histogram to a Lorenz Information Measure (LIM). However, Rorvig teaches converting the area under the histogram to a Lorenz Information Measure (page 14 and 15). Using LIM to convert the area under the histogram is well known in the field of distributional analysis, such as economics.

Therefore, it would had been obvious to one having ordinary skill in the art at the time of the invention was made to use the invention of Rorvig in the combined invention of Eakins, Opittek and Tsymbalenko. One would have been motivated to use Rorvig's invention in the combined invention of Eakins, Opittek and Tsymbalenko because using Rorvig's concept of using LIM on the area of compassed by the histogram will convert histogram values to a single value and then this single value can be encoded as metadata name tags in HTML as per Tsymbalenko rather than attempting to use all of the individual data in the histogram.

Claim 39 has been analyzed and rejected as per claim 29 and in further view of 13.

7. Claims 15, 21-24, 26, 41, 47-50 and 52 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abdel-Mottaleb et al., U.S. Patent No. 5,915,038, in view of Eakins et al., Jan. 1999, "Content-based image retrieval: a report to the JISC technology applications programme", and further in view of Opittek et al., U.S. Patent No. 3,979,555, and further in view of Tsymbalenko et al., Jan. 9, 2001, "Using HTML Metadata to find relevant images on the world wide web".

Claim 15 recites a method of retrieving images by content measure metadata encoding. As explained in the rejection of claim 1, Abdel-Mottaleb, extract (measures) an index key (metadata) for the query (first) image and then generates the key index for each of the images to be searched, where extracting metadata is the measurement of

the features of the still images, and further discloses the selection of the second image with respect to the threshold difference value. Abdel-Mettaleb does not go into the details of constructing histogram for each feature measured, then generating area under histogram and further encoding the areas under histogram in metadata elements of a HTML document.

However, the invention of Eakins combined with Opittek and Tsymbalenko, as explained in the rejection of claim 3, provides the detailed steps of the image analysis required on the query (first) image and all other images to be searched and further provides the concept of encoding of metadata in metadata elements of a HTML document. Therefore, it would had been obvious to one having ordinary skill in the art at the time of the invention was made to use the combined invention of Eakins, Opittek and Tsymbalenko in the invention of Abdel-Mettaleb. One would have been motivated to use the combined invention of Eakins, Opittek and Tsymbalenko in the invention of Abdel-Mettaleb because Abdel-Mettaleb, Eakins, and Tsymbalenko are directed to the methods and systems of image search and retrieval and Eakins further go more in detail of using content features in the image search and retrieval to optimize the search techniques and have better image similarity results and Opittek further provides the concept of image analysis on the selected image features such as obtaining a total mathematical measure (metadata) of color intensity distribution and Tsymbalenko further supports Abdel-Mettaleb method of searching the images on web by encoding the metadata of the images in the metadata elements of the HTML document as explained in the rejection of claim 1.

Claim 21 has been analyzed and rejected as per claim 15 and further in view of claims 3 and 9.

Claim 22 and 23 had been analyzed and rejected as per claim 21.

Claim 24 has been analyzed and rejected as per claim 15 and further in view of claims 3 and 14.

Claim 26 has been analyzed and rejected as per claim 15 and further in view of claim 14, 3 and 1.

Claim 41 recites a system that performs the method recited in claim 15. Claim 41 has been analyzed and rejected as per claim 27.

Claim 47 has been analyzed and rejected as per claim 41 and in further view of claim 21.

Claims 48 and 49 has been analyzed and rejected as per claim 47.

Claim 50 has been analyzed and rejected as per claim 41 and further in view of claim 24.

Claim 52 has been analyzed and rejected as per claim 41 and further in view of claim 26.

8. Claims 16-20 and 42-46 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abdel-Mottaleb et al., U.S. Patent No. 5,915,038, in view of Eakins et al., Jan. 1999, "Content-based image retrieval: a report to the JISC technology applications programme", and further in view of Opittek et al., U.S. Patent No. 3,979,555, and further in view of Tsymbalenko et al., Jan. 9, 2001, "Using HTML Metadata to find relevant images on the world wide web", and further in view of Baxes, 1994, Book Publication, "Digital image processing: principles and applications".

Claim 16 recites "A method of claim 15, wherein measuring selected features further comprises measuring an intensity of a preselected color of the object". Abdel-Mottaleb does not teach of measuring intensity of a preselected color of the object but measuring of intensity of color with motivation has been explained in the rejection of claim. The claim 16 has been analyzed and rejected as per claim 15 and in further view of claim 4.

Claims 17-20 had been analyzed and rejected as per claims 16 and 15 and further in view of claims 5-8.

Claims 42-46 had been analyzed and rejected as per claims as per claim 41 and in further view of claims 16-20.

9. Claims 25 and 51 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abdel-Mottaleb et al., U.S. Patent No. 5,915,038, in view of Eakins et al., Jan. 1999, "Content-based image retrieval: a report to the JISC technology applications programme", and further in view of Opittek et al., U.S. Patent No. 3,979,555, and further in view of Tsymbalenko et al., Jan. 9, 2001, "Using HTML Metadata to find relevant images on the world wide web" and further in view of Rorvig, "An experimental approach for the content-based image analysis: an open source agenda for research".

Claim 25 recites "a method of claim 15, further comprising converting the area under the histogram to a Lorenz Information Measure (LIM)". Abdel-Mottaleb, Eakins, Opittek and Tsymbalenko do not teach converting the area under the histogram to a Lorenz Information Measure (LIM). However, Rorvig teaches converting the area under the histogram to a Lorenz Information Measure (page 14 and 15) as explained in the claim 13 with motivation. Therefore, claim 25 has been analyzed and rejected as per claim 25 and in further view of claim 13.

Claim 51 has been analyzed and rejected as per claim 25.

Conclusion

10. The prior art of record and not relied upon is considered pertinent to applicant's disclosure:

- Kim et al., U.S. Patent No. 6,754,667, discloses a content based image retrieval system and method of retrieving image using the same.
- Tsujimura et al., U.S. Patent No. 5,586,197, discloses image searching method and apparatus thereof using color information of an input image.
- Mukherjea et al., U.S. Patent No. 6,415,282, discloses a method and apparatus for query refinement.
- Yoon et al., U.S. Patent No. 6,621,926, discloses a image retrieval system and method using image histogram.
- Barber et al., U.S. Patent No. 5,579,471, discloses a image content based image query system and method.
- Golshani et al., U.S. Patent No. 6,594,386, discloses a method for computerized indexing and retrieval of digital images based on spatial color distribution.
- Murakawa, U.S. Patent No. 6,463,432, discloses an apparatus for and method for retrieving images.
- Ito et al., U.S. Patent No. 5,555,318, discloses a thresholding method for segmenting gray scale image, method for determining background concentration distribution, and image displacement detection method and also disclose the concept of area under the histogram.

- Stapleton et al., U.S. Patent No. 5,832,140, discloses automated quality assurance image processing system and further provides the concept of area under histogram.
- Wang, U.S. Patent No. 6,373,979, discloses a system and method for determining a level of similarity among more than one image and a segmented data structure for enabling such determination.
- Shimura et al., U.S. Patent No. 5,644,765, discloses a image retrieving method ad apparatus that calculates characteristic amounts of data correlated with and identifying an image.
- Jain et al., U.S. Patent No. 5,893,095, discloses a similarity engine for content-based retrieval of images.
- Jain et al., U.S. Patent No. 5,915,250, discloses a threshold based comparison system and method for image search and retrieval.
- Tang et al., IEEE Publication, 2000, "A content-based image retrieval system on the mode of network". (pp. 422-425)
- Pearce et al., IEEE Publication, 1994, "Theoretical and experimental comparison of the Lorenz Information Measure, Entropy, and the Mean Absolute Error". (pp. 24-29)
- Pass et al., IEEE Publication, 1996, "Histogram refinement for content-based image retrieval". (pp. 96-102)
- Yuwono et al., IEEE Publication, 1996, "Search and ranking algorithms for locating resources on the world wide web". (pp. 164-171)

- Lee et al., IEEE Publication, 1994, "Query by image content using multiple objects and multiple features: user interface issues". (pp. 76-80)
- Pala et al., IEEE Publication, 2000, "Using multiple examples for content-based image retrieval". (pp. 335 –338).
- Newsome et al., IEEE Publication, 1997, "HperSQL: Web-based query interface for biological databases". (pp. 329-339).
- Chen et al., IEEE Publication, 1999, "A synchronized and retrieval video/HTML lecture system for industry employee training". (pp. 750-755).

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Manav Seth whose telephone number is (703) 306-4117. The examiner can normally be reached on Monday to Friday from 8:30 am to 5:00 pm.

If attempts to reach the examiner by telephone are unsuccessful, examiner's trainer, Bhavesh Mehta, can be reached on (703) 305-3885. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you

have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Manav Seth

Art Unit 2625

JOSEPH MANCUSO
PRIMARY EXAMINER

Form PTO-1449 (modified)	Atty. Docket No. 4380.001300/MWS	Serial No. 10/087,347
List of Patents and Publications for Applicant's	Applicant Mark E. Rorvig	
INFORMATION DISCLOSURE STATEMENT	Filing Date: March 1, 2002	Group: 2621
(Use several sheets if necessary)		

U.S. Patent Documents	Foreign Patent Documents	Other Art
See Page 1	*See Page 1*	*See Page 1*

U.S. Patent Documents

Exam. Init.	Ref. Des.	Document Number	Date	Name	Class	Sub Class	Filing Date of App.
	A1						
	A2						
	A3						
	A4						
	A5						

Foreign Patent Documents

Exam. Init.	Ref. Des.	Document Number	Date	Country	Class	Sub Class	Translation Yes/No
	B1						
	B2						
	B3						

Other Art (Including Author, Title, Date Pertinent Pages, Etc.)

Exam. Init.	Ref. Des.	Citation
MW	C1	Rorvig, "Content Based Image Retrieval Enhanced by Integration of Metadata Encoded Image and Text Features," *Texas Center for Digital Knowledge*, The University of North Texas
MW	C2	Jeong et al., "Image Retrieval by Content Measure Metadata Coding," *CIR 2001, Tenth International World Wide Web Conference*, May 1-5, 2001, Hong Kong, International World Wide Web Conference Committee: http://www.iw3c2.org/Conferences/Welcome.html, http://www10.org/cdrom/posters/p1142/index.htm
MW	C3	Goodrum et al., "An Open Source Agenda for Research Linking Text and Image Content Features," *Journal of the American Society for Information Science*, Vol. 52, pp. 948-953 (September 2001), http://www.unt.edu/ir/papers/jasisg00.htm
MW	C4	Rorvig, "An Experimental Approach for the Content-Based Image Analysis: An Open Source Agenda for Research," *The Department of Information Science*, The University of North Texas, pages 1-25

EXAMINER:	DATE CONSIDERED: 01/06/05

EXAMINER: INITIAL IF REFERENCE CONSIDERED, WHETHER OR NOT CITATION IS IN CONFORMANCE WITH MPEP609; DRAW LINE THROUGH CITATION IF NOT IN CONFORMANCE AND NOT CONSIDERED. INCLUDE COPY OF THIS FORM WITH NEXT COMMUNICATION TO APPLICANT.

Form PTO-1449 (modified)	Atty. Docket No. 4380.001300/KDG	Serial No. 10/087,347
List of Patents and Publications for Applicant's INFORMATION DISCLOSURE STATEMENT	Applicant Mark E. Rorvig	
(Use several sheets if necessary)	Filing Date: March 1, 2002	Group: 2621

U.S. Patent Documents	Foreign Patent Documents	Other Art
See Page 1	*See Page 1*	*See Page 1*

U.S. Patent Documents

Exam. Init.	Ref. Des.	Document Number	Date	Name	Class	Sub Class	Filing Date of App.
	A1						
	A2						
	A3						
	A4						
	A5						

Foreign Patent Documents

Exam. Init.	Ref. Des.	Document Number	Date	Country	Class	Sub Class	Translation Yes/No
	B1						
	B2						
	B3						

Other Art (Including Author, Title, Date Pertinent Pages, Etc.)

Exam. Init.	Ref. Des.	Citation
W	C5	Rorvig et al., "Content Based Image Retrieval by Integration of Metadata Encoded Multimedia (Image and Text) Features," *CIR 2002, Eleventh International World Wide Web Conference*, May 6-11, 2002, Honolulu, Hawaii, USA, International World Wide Web Conference Committee: http://www.iw3c2.org/Conferences/Welcome.html, http://www2002.org/CDROM/poster/153.pdf
W	C6	Rorvig et al., "Exploiting Image Primitives for Effective Retrieval," *CIR 2000, Third UK Conference on Image Retrieval*, May 4-5, 2000, Old Ship Hotel, Brighton, UK, Universeity of Brighton: School of Information Management
W	C7	Rorvig et al., "A Common Representation Format for Multimedia Documents," *Texas Center for Digital Knowledge*, School of Library and Information Sciences, University of North Texas, Denton, Texas (2000)
W	C8	Bergman et al., "Progressive Content-Based Retrieval from Satellite Image Archives," *D-Lib Magazine* (October 1997), http://www.dlib.org/dlib/october97/ibm/10li.html

EXAMINER: [signature] | DATE CONSIDERED: 01/06/05

EXAMINER: INITIAL IF REFERENCE CONSIDERED, WHETHER OR NOT CITATION IS IN CONFORMANCE WITH MPEP609; DRAW LINE THROUGH CITATION IF NOT IN CONFORMANCE AND NOT CONSIDERED. INCLUDE COPY OF THIS FORM WITH NEXT COMMUNICATION TO APPLICANT.

Form PTO-1449 (modified)	Atty. Docket No.	Serial No.
List of Patents and Publications for Applicant's	4380.001300/KDG	10/087,347
INFORMATION DISCLOSURE STATEMENT	Applicant: Mark E. Rorvig	
(Use several sheets if necessary)	Filing Date: March 1, 2002	Group: 2621

U.S. Patent Documents	Foreign Patent Documents	Other Art
See Page 1	*See Page 1*	*See Page 1*

Other Art (Including Author, Title, Date Pertinent Pages, Etc.)

Exam. Init.	Ref. Des.	Citation
	C9	Eakins et al., "Content-Based Image Retrieval A Report to the JISC Technology Applications Programme," *Northumbria Image Data Research Institute* (January 1999), http://www.unn.ac.uk/iidr/research/cbir/report.html
	C10	

EXAMINER: [signature] DATE CONSIDERED: 01/06/05

EXAMINER: INITIAL IF REFERENCE CONSIDERED, WHETHER OR NOT CITATION IS IN CONFORMANCE WITH MPEP609; DRAW LINE THROUGH CITATION IF NOT IN CONFORMANCE AND NOT CONSIDERED. INCLUDE COPY OF THIS FORM WITH NEXT COMMUNICATION TO APPLICANT.

Notice of References Cited	Application/Control No. 10/087,347	Applicant(s)/Patent Under Reexamination RORVIG ET AL.	
	Examiner Manav Seth	Art Unit 2625	Page 1 of 3

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
	A	US-5,915,038	06-1999	Abdel-Mottaleb et al.	382/209
	B	US-3,979,555	09-1976	Opittek et al.	348/672
	C	US-6,754,667	06-2004	Kim et al.	707/102
	D	US-5,586,197	12-1996	Tsujimura et al.	382/162
	E	US-6,415,282	07-2002	Mukherjea et al.	707/3
	F	US-6,621,926	09-2003	Yoon et al.	382/168
	G	US-5,579,471	11-1996	Barber et al.	715/700
	H	US-6,594,386	07-2003	Golshani et al.	382/166
	I	US-6,463,432	10-2002	Murakawa, Akira	707/5
	J	US-5,555,318	09-1996	Ito et al.	382/168
	K	US-5,832,140	11-1998	Stapleton et al.	382/298
	L	US-6,373,979	04-2002	Wang, Jia	382/165
	M	US-5,644,765	07-1997	Shimura et al.	707/104.1

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	Tsymbalenko et al., January 2001, "Using HTML Metadata to find relevant images on the world wide web". (pp. 1-9)
	V	Gregory Baxes, 1994, Book Publication, "Digital Image Processing: Principles and Applications".
	W	Tang et al., IEEE Publication, 2000, "A content-based image retrieval system on the mode of network". (pp. 422-425)
	X	Pearce et al., IEEE Publication, 1994, " Theoretical and experimental comparison of the Lorenz Information Measure, Entropy, and the Mean Absolute Error". (pp. 24-29)

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Notice of References Cited	Application/Control No. 10/087,347	Applicant(s)/Patent Under Reexamination RORVIG ET AL.	
	Examiner Manav Seth	Art Unit 2625	Page 2 of 3

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
	A	US-5,893,095	04-1999	Jain et al.	707/6
	B	US-5,915,250	06-1999	Jain et al.	707/100
	C	US-			
	D	US-			
	E	US-			
	F	US-			
	G	US-			
	H	US-			
	I	US-			
	J	US-			
	K	US-			
	L	US-			
	M	US-			

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	Pass et al., IEEE Publication, 1996, " Histogram refinement for content-based image retrieval". (pp. 96-102)
	V	Yuwono et al., IEEE Publication, 1996, "Search and ranking algorithms for locating resources on the world wide web". (pp. 164-171)
	W	Lee et al., IEEE Publication, 1994, " Query by image content using multiple objects and multiple features: user interface issues". (pp. 76-80)
	X	Pala et al., IEEE Publication, 2000, " Using multiple examples for content based image retrieval". (pp. 335-338)

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Notice of References Cited	Application/Control No. 10/087,347	Applicant(s)/Patent Under Reexamination RORVIG ET AL.	
	Examiner Manav Seth	Art Unit 2625	Page 3 of 3

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
	A	US-			
	B	US-			
	C	US-			
	D	US-			
	E	US-			
	F	US-			
	G	US-			
	H	US-			
	I	US-			
	J	US-			
	K	US-			
	L	US-			
	M	US-			

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	Newsome et al., IEEE Publication, 1997, " HyperSQL: Web-based query interface for biological databases". (pp. 329-339)
	V	Chen et al., IEEE Publication, 1999, " A synchronized and retrival video/HTML lecture system for industry employee training". (pp.750-755)
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Using HTML Metadata to Find Relevant Images on the World Wide Web*

Yelena Tsymbalenko
Ethan V. Munson
Dept. of EECS
University of Wisconsin-Milwaukee
Milwaukee, WI 53201 USA
{yelena,munson}@cs.uwm.edu

January 9, 2001

1 Introduction

The World Wide Web has become one of the largest information repositories the world has ever seen. While much of that information is textual, a substantial amount is drawn from other media, especially static images.

An obvious way to use the Web is to treat it as a sort of library that can be indexed, catalogued, and queried. Web search engines and directory-style portals do just that by indexing or cataloguing the Web's textual content to provide convenient access services to end-users.

Providing search services for the Web's image content has been more difficult. A number of researchers have developed Web image search tools, but these systems are limited by the use of visually-based queries and databases that represent a small subset of the Web's content. Companies like Alta Vista and Lycos Multimedia are also beginning to provide image search services. Some features of Alta Vista's image search system suggest that their technology is similar to that reported in this paper.

This paper describes a new approach to finding images on the Web. Instead of analyzing the images themselves using image processing techniques, our software examines the HTML source code that refers to the image and using only this textual information, decides whether or not an image is relevant to a query.

In the next section, we provide some background on prior research into image search and multimedia research using similar strategies. In Section 3, we describe the software testbed we built, while in Section 4 we describe the results of experiments that we ran using this testbed. Section 5 closes with conclusions and suggestions for future research.

*This research was sponsored by the U. S. Department of Defense. Ethan V. Munson was also supported by NSF CAREER award CCR-9734102.

2 Background

There is a large body of research on multimedia indexing and retrieval. Most of this research has been performed using closed databases whose content was under the direct control of the researchers. Examples of such research are easily found in recent conference proceedings [2, 1] and journals [8].

A good example of this course of research is the IBM Almaden Center's Query-By-Image-Content (QBIC) system [6]. QBIC allows users to make image queries based on image features such as shape, color, texture, and object layout. Users define queries either by providing a sample image or by using a graphical tool to make a sketch or diagram. QBIC has a well-developed visual query language and an interesting GUI. Its use of image features for indexing and querying is both an advantage and a disadvantage. When users are seeking images with a particular appearance (e.g. mix of colors, object with a particular shape), it is very helpful. When users are looking for pictures of particular content, it is less helpful because the low-level image characteristics give only limited insight into the real semantics of images. For example, a person's facial appearance may be fairly constant, but their clothing may not be. Many objects (e.g. pencils, fish, or motorcycles) look radically different depending on camera viewpoint. QBIC's approach also appears ill-suited to the scale of the Web. QBIC constructs its indices by pre-analyzing each image in its database. This is computationally demanding and it is difficult to see how it can be done for the Web as a whole.

WebSeek [10] is a more direct attempt to create a directory and database of images from the Web. WebSeek uses a mix of automated and manual techniques to create a database of images downloaded from the Web. It automatically inspects HTML documents, extracting keywords from the image file names that are used to create a histogram of file names. This histogram is used to manually construct a subject hierarchy for the downloaded images. In another manual step, the downloaded images are mapped into the subject hierarchy. Once this is done, WebSeek users can browse the categories in the subject hierarchy, search the categories by keyword, and search the database using image features, especially color histogram information.

WebSeek has a large database of Web images and supports both text-based and image-based queries. Text-based queries have more semantic content than image-based queries. However, it seems unlikely that WebSeek's database can approach the scale of the entire Web, since manual categorization of images is a slow and labor-intensive process.

WebSeer [7] is the system most closely related to this research. The principal investigator, Swain, now works for Alta Vista. Alta Vista has a new image search tool whose qualities appear to derive from Swain's research on WebSeer.

The goal of research on WebSeer was to classify images into categories such as photographs, portraits and computer-generated drawings. To do this, WebSeer supplemented information from image content analysis with information from HTML metadata. WebSeer used several kinds of HTML metadata including the file names of images, the text of the ALT attribute of the IMG tag, and the text of hyperlinks to images to help identify relevant images. Since the WebSeer research emphasized image categorization, this use of metadata is not discussed in detail in any of the WebSeer papers. We assume that the metadata was helpful, but a detailed analysis was not provided.

The research most similar in spirit to that reported in this paper was conducted by Brown et al. [3, 4]. In their first study [3], they used textual "closed captions" transmitted with broadcast news to index stored video. In the second study [4], they used speech recognition techniques to analyze the audio components of video mail. Then, the textual content of the recognized speech was used for indexing the video content. In both studies, Brown et al. took advantage of the fact that data in two media were traveling together and exploited data in one medium to better understand the content in the other.

3 Image Search Architecture

We applied the cross-media indexing strategy of Brown et al. to the Web image search problem. We started with the observation that images on the Web are almost always accessed through HTML documents and that the bulk of the content of HTML documents is textual. In addition, the HTML source includes text that defines a hierarchical information structure. We consider both the textual content and the structure of HTML documents to be "metadata" describing images and use this metadata to determine which images may be relevant to a query.

The second aspect of our strategy was to exploit existing Web search engines in order to search the entire Web, rather than a closed database of previously downloaded images. By using existing search engines, we saved considerable engineering effort and were able to exploit the search engine designers' considerable expertise in computing the relevance of Web documents to textual queries.

We constructed a Web image search application composed of four modules: text search, document download and cleaning, document analysis, and search results interface.

The text search module accepted a one-word query and sent it to the Alta Vista search engine. Alta Vista returned an HTML document with links to ten Web pages that best matched the query. In addition, the bottom of this document had links to as many as nineteen other pages of search results. In effect, Alta Vista returned links to 200 pages having some relevance to our one word queries. The text search module extracted the URLs of these pages from the search results documents and sent a subset of these URLs to the document download and cleaning module.

The download and cleaning module first used the low-level HTTP interfaces to download the Web pages for each URL. In addition, this module downloaded every image referenced by each document, in order to facilitate later analysis. At this point, we confronted the problem that many HTML documents on the Web are ill-formed and thus are difficult to analyze. We solved this problem by using the "Tidy" application [9] developed by Raggett for the Web Consortium. Tidy uses heuristic rules to translate HTML (well-formed or ill-formed) to well-formed XHTML (an analog of HTML that conforms to the XML specification [5]).

The document analysis module parsed the well-formed XHTML documents into an internal tree representation and then searched for "clues" that might indicate that an image in the document matched the query. The analysis module considered an image to match the query if the query appeared in any of the following eight places:

1. An image's file name;
2. The textual content of the document's TITLE element;
3. The value of the ALT attribute of the IMG element;
4. The textual content of an anchor (A) element whose target was the image's file;
5. The value of the TITLE attribute of an anchor (A) element;
6. The textual content of the paragraph that was the parent of the IMG element;
7. The textual content of any paragraph located within the same CENTER element as the IMG element; and
8. The textual content of heading elements that precede the image.

Finally, the search results interface module took the list of matching images generated by the document analysis module and created a Web page interface with links to the matching images and the pages that they came from. This final interface was not designed for end-users, who would certainly prefer an interface based on thumbnail images, but it was suitable for our image search experiments.

At this point, some comments on the design of the testbed are appropriate.

- By using a commercial search engine as the first step in image search, we saved a tremendous amount of engineering effort. However, it clearly makes the set of images returned by the system depend on the behavior of the search engine. At this time, we have no idea what effect the choice of search engine had on our research.
- The eight "clues" used to find matching images were derived from the work on WebSeer and from our own study of the HTML specification and of Web document design practice.
- About 1% of the HTML documents we downloaded were so ill-formed that the Tidy program could not produce an XHTML version.
- We determined that images smaller than 65 pixels in either the horizontal or vertical dimension could be ignored. We found through informal experimentation that such images were essentially always "decorative" elements like borders, bullets, or banner advertisements.

4 Image Search Experiment

Using the testbed described in the previous section, we conducted an image search experiment in the fall of 1999 to assess the effectiveness of our strategy. Our goal was to answer two research questions:

- Which HTML features reveal the most information about images in a document?

- Do image search results depend on the type of query made?

We used our testbed to search for images using twelve one-word queries drawn from five categories. The queries, listed by category, were:

Famous People: "Gorbachev," "Yeltsin," and "Streisand"

Non-famous People: "Yelena" and "Ekaterina"

Famous Places: "Paris" and "London"

Less-famous Places: "Bremen" and "Spokane"

Phenomena: "Explosion," "Sunset," and "Hurricane"

We modified the testbed so that, for each query, it downloaded 30 of the 200 pages returned by Alta Vista and all of the images on those pages. The 30 pages were taken from the first, eleventh, and twentieth search results pages.[1] This procedure could have produced 360 Web pages, but only 276 pages containing a total of 1578 non-decorative images were accessible. For each image, we recorded which of the eight clues would have caused that image to be retrieved by our software. In addition, one of us (Tsymbalenko) looked at each image and classified it as either "relevant" or "not relevant" to the query word.

4.1 Results

We used the human relevance ratings and the data about which images would have been retrieved to compute the standard information retrieval measures of precision and recall. Precision is the proportion of images that a clue caused to be retrieved that are actually relevant to the query. It is computed by the formula

$$\text{Precision} = \frac{\text{Retrieved images that are relevant}}{\text{Total retrieved images}}$$

Recall is the proportion of relevant images (out of the "complete" collection) that are retrieved and computed by the formula

$$\text{Recall} = \frac{\text{Relevant images that were retrieved}}{\text{Total relevant images in collection}}$$

It is important to give a cautionary note about our recall statistics. Recall is normally computed using some standard body of material (e.g. one year's issues of a major newspaper), called a corpus, which is used as the entire "collection" over which searches are performed. Our recall statistics were computed using the 276 HTML documents returned by Alta Vista as our corpus. This is clearly *not* a valid approach, since the set of documents returned by Alta Vista were chosen precisely because they were

[1] We originally chose this approach in the mistaken belief that there would be interesting differences between the first search results (high relevance) and the last search results (low relevance). In retrospect, this was a pointless exercise, because Alta Vista was finding tens of thousands of Web pages that matched our queries, but only providing the best 200 of these. All of these 200 pages were highly relevant to our queries.

Query	Clue							
	1	2	3	4	5	6	7	8
Gorbachev	26.0	84.0	5.0	0.0	0.0	5.0	0.0	0.0
Yeltsin	60.0	60.0	0.0	0.0	0.0	0.0	0.0	0.0
Streisand	23.0	84.6	0.0	0.0	0.0	0.0	0.0	0.0
Yelena	11.1	85.0	5.0	0.0	0.0	0.0	0.0	0.0
Ekaterina	0.0	60.0	14.3	0.0	0.0	0.0	0.0	0.0
Paris	62.0	62.0	0.0	0.0	0.0	0.0	0.0	0.0
London	12.5	95.8	12.5	0.0	0.0	0.0	0.0	0.0
Bremen	80.0	90.0	33.0	0.0	0.0	0.0	0.0	0.0
Spokane	90.0	100.0	30.0	0.0	0.0	0.0	0.0	0.0
Explosion	25.0	75.0	0.0	0.0	0.0	0.0	0.0	0.0
Sunset	88.8	11.1	0.0	0.0	0.0	0.0	0.0	0.0
Hurricane	53.8	38.5	0.0	0.0	0.0	0.0	0.0	0.0
Median percent	39.9	79.5	2.5	0.0	0.0	0.0	0.0	0.0

Table 1: Recall percentages for each clue and each query.

highly relevant to our query and this could easily bias our results. Unfortunately, we know of no standard Web corpus on which to perform our tests. Thus, we use our recall statistics only to give initial results on the relative merits of our metadata clues, not to make comparisons to other search approaches.

Our recall results are shown in Table 1. Only clues 1 (image file name) and 2 (content of document's TITLE element) show high levels of recall with medians of 39.9% and 79.5%, respectively. Clue 3 (value of the ALT attribute of the IMG element) shows a modest level of recall with a median of only 2.5%, but individual recall percentages as high as 33%. Only one other clue, number 6 (textual content of a paragraph that contains an IMG element), showed any recall at all.

Precision results are shown in Table 2. For clue 1 (image file name), precision ranges from 36% to 100%

4.2 Discussion

Examining the results of the image search experiment closely, several key results emerge.

The three clues (1, 2, and 3) that show significant levels of recall are relatively simple. Image file name (clue 1) presumably works because Web site designers prefer mnemomic names for image files. The TITLE element (clue 2) is designed to provide a high-level description of a document's content and is widely used because it get listed in search engine results and the browser's title bar. The ALT attribute of the IMG element (clue 3) is explicitly designed to be a textual alternative to the image itself. The remaining five clues generally emphasize HTML's underlying structural model, and based on our results, do not seem to be widely used idioms among HTML authors and showed essentially no recall.

Looking at the types of queries, image file name (clue 1) had poor recall for the

Query	Clue							
	1	2	3	4	5	6	7	8
Gorbachev	83.0	46.0	100.0	—	—	100.0	—	—
Yeltsin	100.0	60.0	—	—	—	—	—	—
Streisand	100.0	47.8	—	—	—	—	—	—
Yelena	66.7	89.5	100.0	—	—	—	—	—
Ekaterina	—	100.0	100.0	—	—	—	—	—
Paris	84.0	70.0	—	—	—	—	—	—
London	60.0	46.9	75.0	—	—	—	—	—
Bremen	66.7	69.2	100.0	—	—	—	—	—
Spokane	75.0	71.4	100.0	—	—	—	—	—
Explosion	100.0	50.0	—	—	—	—	—	—
Sunset	36.0	50.0	—	—	—	—	—	—
Hurricane	100.0	35.7	—	—	—	—	—	—
Median percent	83.0	55.0	87.5	0.0	0.0	0.0	0.0	0.0

Table 2: Precision percentages for each clue and each query. Dashes are used for clue-query combinations that had zero recall, since precision cannot be computed when there is no recall. Median precision percentages are computed based only on those queries that had some recall.

names of people, but excellent recall for place name queries, particularly the less famous cities. An informal look at the details of this phenomenom suggests that Web designers often use nicknames for the image file names people (e.g. "Gorby" for "Gorbachev"), but usually use full names for places.

The precision results are more striking. The three queries that had some recall all showed good precision. The image file name had precision ranging from 36% to 100% with a median of 83%. The content of the TITLE element had precision ranging from 35.7% to 100% with a median of 55%. These precision results are strong by the normal standards of textual information retrieval. The precision results for the value of the ALT attribute of the IMG entity are quite impressive. In general, this clue had 100% precision.

Now, it is possible to examine the original research questions that we posed.

First, we asked which HTML features revealed the most information about the images in a document. It is clear that only three of the HTML features that we tested showed any real utility for identifying the content of images. Image file name, the content of the TITLE element, and the value of the ALT attribute appear useful in image search, while the other clues we tested do not appear useful.

Second, we asked whether the type of query affected our image search results. The type of query does seem to affect our recall results, where the names of people show less recall than the names of places. No consistent effect can be seen in the precision results.

4.3 Cautionary Notes

These results should be viewed cautiously. This was a small study and it has some flaws.

- Our results are affected by our use of the Alta Vista search engine. It is not clear what effect this had, but the use of a different search engine might produce different results.
- Our relevance ratings for the images were performed by one person. They should really be based on the judgement of multiple relevance raters. Also, image relevance is harder to judge than text relevance and may require somewhat different rating methods.
- The distinction between famous and non-famous people is confounded with another effect. All of the famous names used were family names. Both of the non-famous names used were personal names. Also, one of the non-famous names is actually the personal name of moderately famous person (the skater Ekaterina Gordeeva).
- The one word queries we used do not allow the construction of very precise queries, especially for the names of people.
- Our use of the Tidy program may have removed some clues. We believe that an author writing HTML like the following, probably views the image as part of the first paragraph (that is, a child of the paragraph). Tidy makes the image a sibling.

```
<P>Some text. <IMG href="img.gif">
<P>More text.
```

5 Conclusion and Suggestions for Future Research

This paper has described new software for finding images on the Web based on simple text queries and an experiment testing the techniques used by that software. The software used a text search engine to find documents containing text matching the query and then analyzes the content of the document to determine whether the images in that document may be relevant to the query. The experiment demonstrated that some of the techniques used to identify relevant images were effective and that others were not. Its results also suggested that the type of query made alters the effectiveness of the search technique.

Why is this software interesting? Image search is an inherently interesting problem and is being studied widely. This software is interesting because it is able to find relevant images without actually downloading or analyzing those images. Instead, it examines only the text that surrounds the reference to the image in the HTML document. It is widely known that image download requires substantial amounts of time when traversing the Web. Any system that can find images without downloading them

has an inherent performance advantage over systems that must download images. Furthermore, we believe that the text in an HTML document gives more precise semantic information about the content of images than any existing image processing technique. Image processing can be used to determine that an image shows the face of a person, but the file name of the image may say exactly what person is shown.

Considerably more research is called for. Our basic results showing the success of our technique need to be replicated using a larger study, more complex queries, and a more robust relevance rating system. While our techniques appear strong and they have an efficiency advantage over image processing approaches, there is no reason that textual metadata cannot be combined with image processing in order to produce even better results. Finally, we continue to believe the type of query will interact with search heuristics in interesting ways.

References

[1] *Proceedings, ACM Multimedia '00*. ACM Press, November 2000.

[2] *Proceedings, ACM Multimedia '99*. ACM Press, November 1999.

[3] M. G. Brown, J. T. Foote, G. J. F. Jones, K. Sparck Jones, and S. J. Young. Automatic content-based retrieval of broadcast news. In *Proceedings ACM Multimedia '95*, pages 35–44. ACM Press, November 1995.

[4] M. G. Brown, J. T. Foote, G. J. F. Jones, K. Sparck Jones, and S. J. Young. Open-vocabulary speech indexing for voice and video mail retrieval. In *Proceedings ACM Multimedia '96*, pages 307–316. ACM Press, November 1996.

[5] World Wide Web Consortium. Extensible markup language (xml) 1.0. Available on the Web at http://www.w3.org/TR/2000/REC-xml-20001006, October 2000. Second edition.

[6] Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, David Steele, and Peter Yanker. Query by image and video content: the QBIC system. *Computer*, 28(9):23–32, September 1995.

[7] Charles Frankel, Michael Swain, and Vissilis Athitsos. WebSeer: An image search engine for the World Wide Web. Technical Report 96-14, University of Chicago, Department of Computer Science, July 1996.

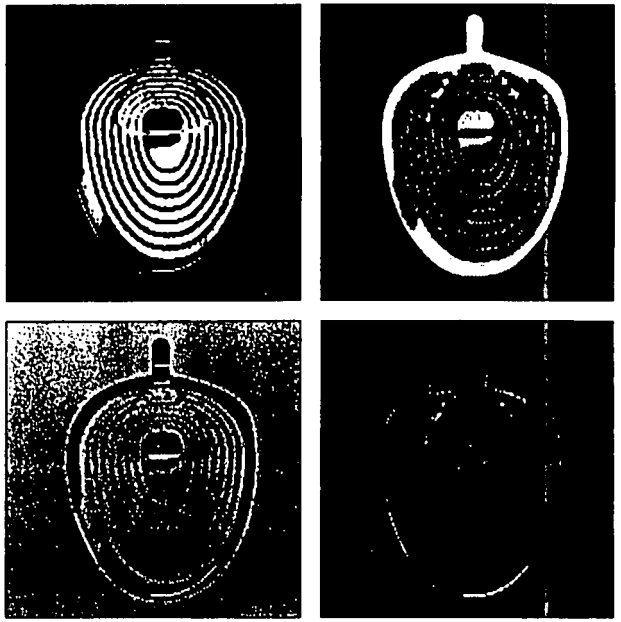
[8] *Multimedia Systems*. Published jointly by Springer Verlag and ACM Press. Quarterly academic journal.

[9] Dave Raggett. Clean up your web pages with HTML TIDY. Available at www.w3.org, August 2000. Version of August 4, 2000.

[10] J. Smith and S. F. Chang. Visually searching the Web for content. *IEEE Multimedia*, 4(3):12–20, July-September 1997.

DIGITAL IMAGE PROCESSING

■ PRINCIPLES AND APPLICATIONS ■



GREGORY A. BAXES

About the cover:

The front cover images are digitally processed microscope views of a thin-film technology recording head. These miniaturized coils are part of the read/write head subsystem used in IBM's computer hard disk drives. Each image was processed using a different brightness slicing operation and then false-colored to highlight the internal spiral structure. Subsequent machine vision operations can follow these operations to automatically measure the important pattern dimensions, freeing the human from the tedious task while improving the inspection accuracies two- or three-fold. These images appear courtesy of International Business Machines Corporation.

Library of Congress Cataloging-in-Publication Data

Baxes, Gregory A.
Digital image processing : prinicples and applications / Gregory A. Baxes.
p. cm.
Includes index.
ISBN 0-471-00949-0 (paper)
1. Image processing—Digital techniques I. Title.
TA1637.B39 1994
621.36'7—dc20

94-9744
CIP

Printed in the United States of America
10 9

To my friend and colleague
Bert Kadzielawa
your memory lives forever

In motion image sequences, interlaced scan displays can show noticeable motion defects because the odd and even halves of each image are separated in time by one-half the frame rate. The result is a tearing effect that appears on objects with a fast rate of motion through the image frame.

The *non-interlaced* method of image display is known as *progressive scan* display. Progressive scan means that the entire image is displayed in one pass. In this case, the frame rate must be twice that of an equivalent interlaced display, or image flickering will be noticeable. Progressive scan eliminates line-to-line flicker and motion artifacts in displayed images. Systems using a progressive scan display monitor for image display typically have a 72 frame-per-second frame rate.

Qualities of the Digital Image

To determine appropriate processing steps, it is often desirable to assess the overall qualities of an image. Of particular interest are the qualities that describe the brightness and spatial frequency characteristics. Two important tools, the brightness histogram and spatial frequency transforms, can uncover telltale signs of an image's overall deficiencies and strengths.

Brightness Histograms

The brightness characteristics of an image can be concisely displayed with a tool known as the *brightness histogram*. In general terms, a histogram is a distribution graph of a set of numbers. The brightness histogram is a distribution graph of the gray levels of pixels within a digital image. It provides a graphical representation of how many pixels within an image fall into the various gray-level steps.

A histogram appears as a graph with "brightness" on the horizontal axis from 0 to 255 (for an 8-bit gray scale) and "number of pixels" on the vertical axis. To find the number of pixels having a particular brightness within an image, we simply look up the brightness on the horizontal axis, follow the bar graph up, and read off the number of pixels on the vertical axis. Because all pixels must have some brightness defining them, the number of pixels in each brightness column adds to the total number of pixels in the image.

The histogram gives us a convenient, easy-to-read representation of the concentration of pixels versus brightness in an image. Using this graph we are able to see immediately whether an image is basically dark or light and high or low in contrast. Furthermore, it gives us our first clues about what contrast enhancement would be appropriately applied to make the image more subjectively pleasing to an observer, or easier to interpret by succeeding image analysis operations.

Contrast and Dynamic Range Indications

Contrast is a term that is often used to describe the brightness attributes of an image. We intuitively understand contrast to mean how vivid or washed-out an image appears with respect to gray tones. Contrast in an image is clearly illustrated in the brightness histogram. Low contrast appears as a tightly grouped mound of pixel brightnesses in the gray scale, leaving other gray levels minimally or completely unoccupied. High contrast appears as a bimodal histogram—two peaks exist at the outer brightness regions, leaving the gray levels in between unoccupied.

The histogram also tells us how much of the available dynamic range is used by an image. The actual dynamic range of an image is represented by how many gray levels in the gray scale are occupied. For instance, a mound of pixels falling between gray values 50 and 100 (within a range of 0 to 255), with none in the other regions, indicates a small dynamic range of brightness, whereas a wide gray-scale distribution shows large dynamic range. An image with small dynamic range does not occupy all the available spread of gray levels; the image has really only been quantized to a gray scale composed of the occupied range. This indicates low brightness resolution, along with low contrast. Large dynamic range generally implies a well-balanced image, except if it is a bimodal distribution, in which case the image is high contrast. Three common histograms, along with their original images, are illustrated in Figure 3.19.

Color Histograms

Histograms for color images are a threefold version of the regular, gray-level brightness histogram. Three histograms are computed and displayed, one for each color component, as shown in Figure 3.20. The color histograms can represent

***Figure 3.19a** Image with low contrast and low dynamic range.*

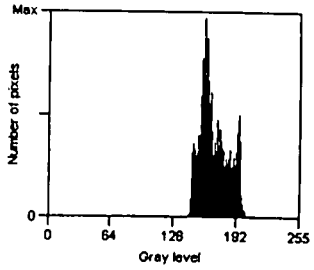


***Figure 3.19b** Low contrast and low dynamic range histogram.*

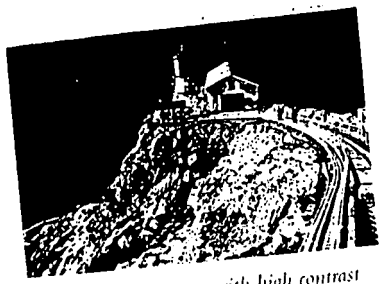


Figure 3.19c Image with high contrast and high dynamic range.

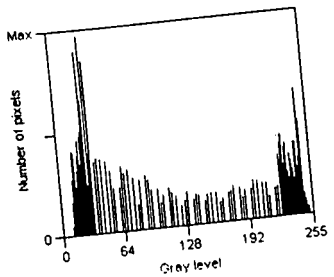


Figure 3.19d High contrast and high dynamic range histogram.

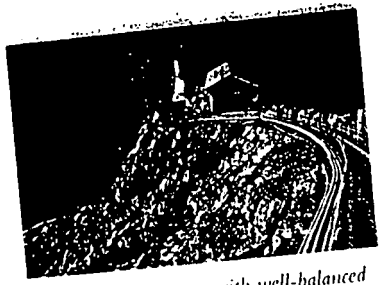


Figure 3.19e Image with well-balanced contrast and high dynamic range.

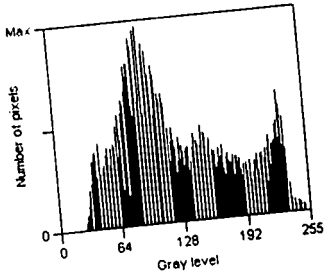


Figure 3.19f Well-balanced contrast and high dynamic range histogram.

RGB space, HSB space, or any other color space. Each histogram can help determine the brightness distributions, contrast, and dynamic ranges of the individual color components.

Spatial Frequency Transforms

Earlier we discussed the concept of image spatial frequency content. We that details within an image comprise low or high spatial frequency comp depending on the rates at which the details transitioned from dark to li back to dark. *Frequency transforms* provide a pictorial view of these spatial fr components. The most common frequency transform is the Fourier t however, many others also exist.

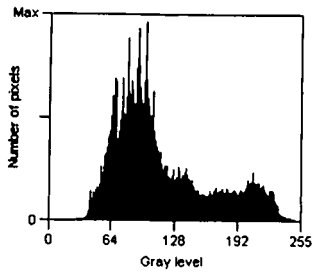


Figure 3.20a The red component histogram of Figure 3.15a image.

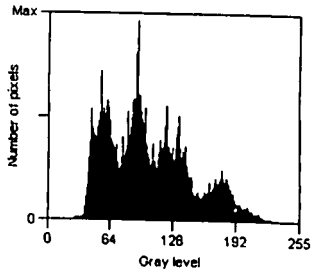


Figure 3.20b The green component histogram of Figure 3.15b image.

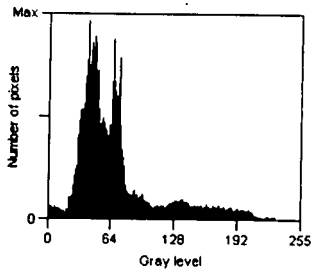


Figure 3.20c The blue component histogram of Figure 3.15c image.

Essentially, a digital image is made up of fundamental spatial frequency components that, when combined, make up the form of the image. These two-dimensional spatial frequency components have varying orientations—horizontal, vertical, diagonal, and so on. A frequency transform converts an image from the spatial domain of brightnesses to the frequency domain of frequency components. This frequency decomposition results in a new image that displays the array of spatial frequency components present in the original image, as shown in Figure 3.21.

The frequency image shows the original image's spatial frequency components by the brightness of points at respective locations. "Horizontal frequency" is defined along an imaginary x-axis that passes horizontally through the center of the image. Likewise, "vertical frequency" is defined along an imaginary y-axis that passes vertically through the center of the image. Diagonal frequencies are all the points in between. The brightness of a point in the frequency image corresponds to the magnitude of the frequency component represented by the point's coordinates. The negative frequencies—those to the left of the y-axis and below the x-axis—are merely a mirroring of the positive frequencies, and are therefore not relevant in our quality assessment.

inal image. When the numeric range of a pixel's brightness is increased, so is the pixel's brightness resolution.

Digressing momentarily, let's clarify our definitions of the terms *intensity* and *brightness*. "Intensity" (or, more appropriately, *radiant intensity*) refers to the magnitude, or amount, of light energy actually reflected or transmitted from a physical scene. The term "brightness" (or, more appropriately, *luminous brightness*) refers to the measured intensity after it is acquired (say, using a video camera), sampled, quantized, displayed, and observed (with our eyes). The brightness of a pixel accounts for all the effects induced by the entire imaging system. This may seem like a somewhat trivial distinction, but it accounts for the fact that the measured brightnesses in our digital image are only representations of the actual energy radiated from the original physical scene. Additionally, the terms "intensity," "brightness," "radiance," and "luminance" are often used synonymously to mean the same thing. They are, however, distinctly different terms relating to measures of the quantity of light. As we discuss digital image processing operations, we will refer to digital pixels as having a brightness property.

Following the sampling process, each sample is quantized. This quantization process converts the continuous-tone intensity, at the sample point, to a digital brightness value. The accuracy of the digital value is directly dependent upon how many bits are used in the quantizer. If three bits are used, the brightness can be converted to one of eight gray levels. In this case, gray level "0" represents black, gray level "7" represents white, and gray levels "1" through "6" represent the ascending gray tones between black and white. The eight gray levels comprise what is called the *gray scale*, or in this case, the 3-bit gray scale.

With a 4-bit brightness value, every pixel's brightness is represented by one of 16 gray levels. A 5-bit brightness value yields a 32-level gray-scale range. An 8-bit brightness value yields a 256-level gray-scale range. Every additional bit used to represent the brightness doubles the range of the gray scale. The range of the gray scale is also referred to as *dynamic range*. An image with 8-bit brightness values is said to have an available dynamic range of 256 to 1. Several gray scales are shown in Figure 3.10. Notice how the smoothness of the gray scale improves as more bits are used to represent brightnesses.

Figure 3.11 shows an image quantized to various brightness resolutions. The image quantized to eight bits of brightness resolution appears very natural and continuous. As the brightness resolution decreases, the image appears coarser and more mechanical. This effect is known as *brightness contouring*, or *posterization*. Contouring occurs when there are not enough gray levels to represent the actual brightness in the original image adequately. Gradual brightness changes end up quantized a little brighter or dimmer than their original intensities. Brightness contouring is the effect of insufficient brightness resolution.

Figure 3.12 shows the same image broken into individual *bit-planes*. Each bit-plane represents the on or off level of the particular bit's contribution to the overall pixel brightness. Clearly, much of the structure of the image is conveyed

A Content-based Image Retrieval System on the Mode of Network

Hongmei Tang Ming Yu Zhitao Xiao Yingchun Guo
Hebei University of Technology, Tianjin, 300130, China
E-mail: yuming@hebut.edu.cn

Abstract

The paper presents a small image retrieval system on the mode of computer network. In order to overcome some limitations of the traditional retrieval methods, we design an image retrieval system that integrates text-based and content-based image retrieval (CBIR) techniques. When the user submits the query, the system performs the query programs. The retrieval results according to the similarity to the queried image are shown to the user through Web browser. Experiments show that this system is simple, practical, effectively and can be extended easily.

I. Introduction

With the coming of information society and the widely applications of Internet technology, people meet with more and more images and videos frequently. It becomes an urgent task that how to organize, manage and retrieve these information efficiently. Image retrieval is one of the most important works in image processing and machine intelligence. It has been used widely in digital library, remote teaching, remote medical, trademark management, security system and material analysis *etc.*

The text-based approach relies on text description of image contents and makes use of the traditional information retrieval (IR) techniques. Text-based techniques can capture high level abstraction and concepts. It is easy to issue text queries. But text descriptions are sometime subjective and incomplete, and cannot depict complicated image features very well. Text-based techniques cannot accept pictorial queries (query by example).

In the recent years, content-based image retrieval techniques have been proposed to overcome the limitations of the text-based retrieval techniques [1-3]. CBIR (content-based image retrieval) technology means the low-level visual features of images such as color, shape, texture, contour and location can be used as image content table to match and retrieval image. Content-based techniques can capture low-level image features and accept pictorial queries. But they cannot capture high-level concepts. Pictorial query process is hard to start, as the user has to specify the query image by selecting an existing image or drawing a sketch.

We propose to integrate the text-based and content-based techniques into one system. Such a system can capture both high and low level features. Users can start their search process by issuing a text query. From the initial returned images, they can select images as content-based queries. The final returned images are based on combined matching scores of the text-based and content-based searching, incorporating the user's preference weighting. The system is easy to use because the user starts a retrieve process by typing in keywords instead of having to use example images or to describe image features.

There are many content-based image retrieval techniques, such as those based on color, texture and shape [1-3]. In our prototype system, we implemented the color based retrieval technique for the following reasons. Firstly, the color-based technique has been reported to produce good retrieval performance. Secondly, it is simple to implement. Unlike the texture based and shape based methods, it does not require image segmentation which itself is a hard image processing problem.

Another image analysis about object's symmetry is also included in our image retrieval system. Symmetry is an important mechanism by which we identify the structure of objects. A limited number of approaches have been tried in the detection of symmetry in images. The symmetry analysis has been resolved by the local energy model and phase congruency in the system.

In the following section, we describe the details of the

0-7803-6253-5/00/$10.00 ©2000 IEEE.

proposed integrated image retrieval system. We then present some experimental results in Section 3. Section 4 concludes the paper.

II. The Image Retrieval System

2.1 The image retrieval system architecture

The Web browser/Web server (B/S) mode has been adopted in our image retrieval system. The outstanding character of this management mode lies in that programs, database and other component are all concentrated on the server. The maintenance and updating of the system, the backup of database and daily maintenance *etc* are all finished in the server. The browser doesn't need any other software, correlative management and maintenance tasks. Thus, the data that the users retrieve and operate are all from the same database which ensure data's integrity. Client and server interact by common gateway interface (CGI) or active server page (ASP). Generally speaking, when the Web server application on working, according to the standard http, the 80 port (other ports are within the consideration, which depends on how the server is set) will listening to the request from the users. When the users send their requests through browser, the application can carry out the response according to the request of users after it obtains the parameters offered by the users. In the end, it sends the final result to the users and completes the interact course in this way. The users can do database operations quickly and conveniently by way of the universal client tool —— browser. Our image retrieval system architecture is showed in Figure 1.

2.2 The integrated image retrieval process

We collect images and HTML documents, creates indexes based on text description and color histograms of images firstly. And then during the retrieval phase, the system calculates the similarity between the query and indexed images based on a combination of text and color indexes. We can carry out text-based image indexing and retrieval based on text description within HTML documents using the traditional text information retrieval (IR) techniques [4]. Different HTML documents are created by different authors. Different forms of the same words may be used in different documents. Words or terms appearing at different locations of an HTML document have different levels of importance or relevance to related images. For example, a word appearing in the image URL is more important to the image than another word appearing in a paragraph somewhere else in the document. Therefore, in order to improve the retrieval effectiveness, we have assigned term weights based on term positions: higher weights are assigned to terms that are considered more important based on their appearing positions.

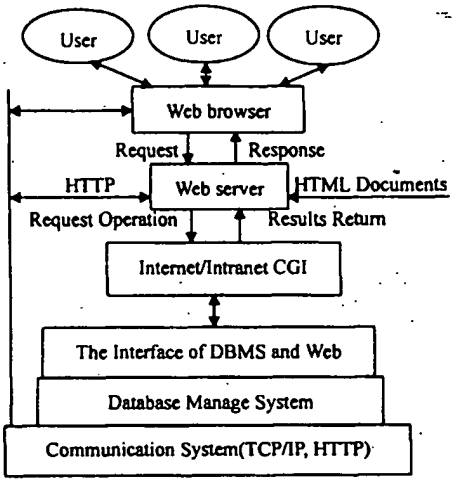


Fig.1 The Architecture of Image Retrieval System

In the color based image retrieval technique, each image in the database is normally represented using three primaries of the color space chosen. Each color channel is quantized into m intervals. So the total number of discrete color combinations (called bins) n is equal to m^3. A color histogram H(M) is a vector (h1, h2,..., hn), where each element hj represents the number of pixels falling in bin j in image M. These histograms are the feature vectors (indexes) to be stored as the index of the image database. During image retrieval, a histogram is found for the query image or estimated from the user's query. A metric is used to measure the distance between the histograms of the query image and images in the database. If images are of different size, their histograms are normalized. Images with a distance smaller than a pre-defined threshold are retrieved from the database and presented to the user according to the similarity to the queried image. The main

problem with the basic color based image retrieval is that a color may be similar to colors of more than one bin, but it is normally only quantized into one bin, leading to the problem that images consisting of similar colors have very large histogram distance. To solve the above problems, instead of dividing each color channel by a constant (quatization step) when obtaining histogram, we propose to find representative colors in the CIEL*u*v* color space. The number of the representative colors is equal to the required number of bins. These representative colors are uniformly distributed in the CIEL*u*v* color space. While building histogram, ten perceptually most similar representative colors are found for each pixel. Then weights are assigned to these ten representative colors inversely proportional to the color distances. The total weights for each pixel is equal to 1. In this way, we obtain a so-called perceptually weighted histogram (PWH) for each image. It has been shown that the PWH based method has higher image retrieval performance than the normal color histogram based method [5].

The user starts a retrieval process by typing in some key words. Once the initial result is presented to the user, the user can select an image to do a search based on its PWH, Alternatively, the user can make a query based on a combination of text and PWH. Figure 2 shows the retrieval results based on key words and the PWH of the selected image.

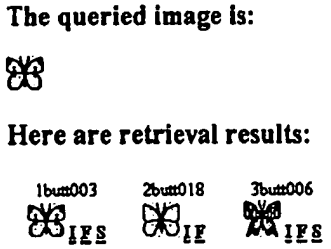


Fig.2. The Image Retrieval Results

We can conclude that different query combinations result in different images and presentation orders. The user can choose an appropriate query combination based on his/her requirements and knowledge. For example, the user can assign a high weight to text if he/she is looking for some high level concepts. On the other hand, if he/she is looking for something with similar color distribution, he/she can assign high weights to PWH. The retrieval results are shown to the user according to the similarity to the queried image through Web browser. There are two or three hyper linked buttons under every image. Click these buttons, you can get the corresponding image information (such as the detailed information about this image, display the full image *etc*).

Another image analysis about object's symmetry is also included in our image retrieval system. Symmetry is an important mechanism by which we identify the structure of objects. Man-made objects, plants and animals are usually highly recognizable from the symmetry, or partial symmetries that they often exhibit. A limited number of approaches have been tried in the detection of symmetry in images. A fundamental weakness found in most is that they require objects to be segmented prior to any symmetry analysis. For example Atallah [6] describes an algorithm that requires objects to be represented in terms of points, line segments and circles. A new measure of symmetry that is presented that does not require any prior recognition or segmentation of objects by Poter Kovesi [7]. This measure is based on phase congruency and local energy model. The phase congruency function is developed from the Fouries series expansion of one dimensional signal, I at some lacation, x, $I(x)=\sum_n A_n \cos(n\omega x+\phi_{n0})=\sum_n A_n \cos(\phi_n(x))$, where A_n represents the amplitude of the n^{th} cosine component, ω is a constant (usually 2π), and ϕ_{n0} is the phase offset of the n^{th} component. The function $\phi_n(x)$ represents the local phase of the Fourier component at position x. Phase congruency is defined as $PC(x)=\max_{\bar{\phi}(x)\in[0,2\pi]} \frac{\sum_n A_n \cos(\phi_n(x)-\bar{\phi}(x))}{\sum_n A_n}$. The value of $\bar{\phi}(x)$ that maximizes this equation is the amplitude weighted mean local phase angle of all the Fourier terms at the point being considered. As phase congruency is a rather awkward to calculate. As an alternative to this

Venkatesh and Owens [8] show that points of maximum phase congruency can be calculated equivalently by searching for peaks in the local energy function. The local energy function is defined for a one dimensional luminance profile, $I(x)$, as the modulus of a complex number, $E(x)=\sqrt{I^2(x)+H^2(x)}$, where the real component is represented by $I(x)$ and the imaginary component by $iH(x)$, where $i=\sqrt{-1}$ and $H(x)$ is the Hilbert transform of $I(x)$. Venkatesh and Owens prove that energy is equal to phase congruency scaled by the sum of the Fourier amplitudes, that is $E(x)=PC(x)\sum_n A_n$. Phase congruency is calculated via wavelet. Finally we can get a measure of symmetry:

$$Sym(x)=\frac{\sum_n \lfloor A_n(x)\lfloor|\cos(\phi_n(x))|-|\sin(\phi_n(x))|\rfloor-T\rfloor}{\sum_n A_n(x)+\varepsilon}$$, where

ε is a small constant to avoid division by zero, T is the estimated noise influence, and $\lfloor\ \rfloor$ denotes that the enclosed quantity is itself if it is positive, and zero for all other values. One dimension analysis can be extended to two dimensions. The measure of symmetry is normalized, dimensionless measure. They are independent of the brightness or contrast of image features.

III. Experiment Results

The image retrieval system was tested on text searching, color searching and combined searching of text and color. Three queries for each set were performed on a database of approximately 800 images. The experiment results show that the combined retrieval returns results that matched either the semantic meaning of terms or the low-level color features of images or both.

IV. Conclusions

In this paper, we described an integrated image retrieval system. The system is based on the improved text-based and color-based image retrieval techniques. Our experimental results obtained from a database of 800 images show that the integrated system has higher retrieval performance than the text based and the color based techniques. The system is also easy to use as it allows the user to start a query by typing in keywords. Once having queries by combining keywords and example images with varying weights. Another image analysis about object's symmetry is also included in our image retrieval system. The system is stable, convenient and can be extended easily. The system realizes organization, management and retrieval of the image database under B/S mode.

Acknowledgments

This work is supported by Hebei Province Natural Science Fund Committee (600056).

References

[1] Niblack W. et al, "QBIC Project: querying images by content, using color, texture, and shape", Proceedings of Conference on Storage and Retrieval for Image and Video Databases, 1993, California, US, SPIE Vol. 1908, pp.1908-1920.

[2] Swain M. J. and Ballard D. H., "Color indexing", Int. J. Comput, Vision 7, pp.11-32.

[3] Stricker M. and Orengo M., "Similarity of color images", Proceedings of Conference on Storage and Retrieval for Image and Video Database 1995, California, SPIE Vol. 2420, pp. 381-392.

[4] Chua T. S et al, "A Concept-based Image Retrieval System", Proceedings of 27th Annual Hawaii International Conference on System Science, Maui, Hawaii, January 4-7 1994, Vol. 3, pp. 590-598.

[5] Lu G. and Phillips J., "Using perceptually weighted histograms for color-based image retrieval", Proceedings of Fourth International Conference on Signal Processing, 1998, China, pp. 1150-1153.

[6] J.R.Atallah. "On symmetry detection". IEEE Transactions on Computers, 1985, C-34, pp. 663-666.

[7] Peter Kovesi. "Invariant Measures of Image Features from Phase Information", The Degree of Doctor of Philosophy of the University of Western Australia, 1996, pp.33-60.

[8] S.Venkatesh and R.A.Owens. "An energy feature detection scheme". In the international conference on image processing, 1989. Singapore, pp.553-557.

Theoretical and Experimental Comparison of the Lorenz Information Measure, Entropy, and the Mean Absolute Error

Tom McMurray John A. Pearce

Biomedical Engineering
The University of Texas at Austin
Austin, TX 78712

Abstract—**The Lorenz information measure (LIM) is a function of the observed probability sequence of digital signals, similar to the signal entropy, and is approximately linearly related to the mean absolute error (MAE) in simulations employing uncorrupted and corrupted 2-dimensional gaussian and magnetic resonance (MR) images. Unlike the MAE, the LIM does not require an uncorrupted reference signal for a distance computation. However, for the particular difference signal case imposed by the definition of the MAE, the LIM is asymptotically bounded by the MAE/signal quantization number ratio. Therefore, in applications where an uncorrupted signal does not exist, and thus, the MAE is undefined, the LIM provides a comparable signal processing performance measure.**

I. Introduction

The mean absolute error (MAE) is a metric commonly used in comparing signal processing algorithms. Accordingly, this computation requires an uncorrupted reference signal from which the distance to the observed signal is determined. However, if the reference signal does not exist or cannot be measured, the MAE is undefined. Thus, an alternate measure for signal processing performance is necessary in the absence of a reference signal. The Lorenz information measure (LIM) is a function of the observed probability sequence of a digital signal, similar to the signal entropy, providing performance comparable to the MAE without the requirement of an uncorrupted signal.

A. Mean Absolute Error

Consider subsets of the l^1 metric space consisting of finite sets of finite valued sequences $a = (a_0, a_1, \ldots, a_{m-1})$, representing the reference signal, and $\hat{a} = (\hat{a}_0, \hat{a}_1, \ldots, \hat{a}_{m-1})$, representing the corrupted signal, with the convergence condition provided by the finite l^1 norm and the distance given by [1-5]:

$$\| (\hat{a} - a) \|_{l^1} = \sum_{i=0}^{m-1} |\hat{a}_i - a_i| \qquad (1)$$

where $\hat{a} = a + \nu$ and ν is the additive noise. Assuming that the originally real-valued sequences are uniformly quantized into at most n discrete levels for standard digital signal processing applications, the resulting digital signals may be scaled to represent arbitrary rational numbers and are, thus, considered integers to facilitate the analysis without loss of generality. The MAE is the metric given by:

$$\frac{1}{m} \| (\hat{a} - a) \|_{l^1} = \frac{1}{m} \sum_{i=0}^{m-1} |\hat{a}_i - a_i| \qquad (2)$$

B. Lorenz Information Measure

Continuing with this difference signal structure imposed by the definition of the MAE for the development of the Lorenz information measure, define the histogram $h = (h_0, h_1, \ldots, h_{n-1})$ as the number of elements in $|\hat{a} - a|$ with the value j where $0 \leq j \leq n-1$. Scaling the elements of h by m, the total number of elements in $|\hat{a} - a|$, produces a probability sequence given by:

$$\begin{aligned} p &= \left(\frac{h_0}{m}, \frac{h_1}{m}, \ldots, \frac{h_{n-1}}{m}\right) \qquad (3) \\ &= (p_0, p_1, \ldots, p_{n-1}) \\ 0 &\leq p_j \leq 1 \ \forall j \\ \sum_{j=0}^{n-1} p_j &= 1 \end{aligned}$$

0-8186-6250-6/94 $3.00 © 1994 IEEE

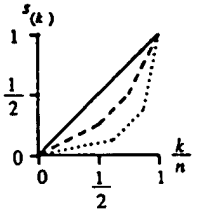


Figure 1. A family of Lorenz curves are illustrated with a uniformly distributed signal represented by the linear solid line and signals approaching a constant depicted by the dashed lines.

equivalent to the sequence required to compute the signal entropy H, according to:

$$H = -\sum_{j=0}^{n-1} p_j \log_2(p_j) \tag{4}$$

and the minimum quantization number 2^H [6]. Arranging the sequence in ascending order results in a monotonically nondecreasing sequence $p_{(j)} = (p_{(0)}, p_{(1)}, \ldots, p_{(n-1)})$, such that $p_{(0)} \leq p_{(1)} \leq , \ldots, \leq p_{(n-1)}$. Graphing the partial sums $s = (s_{(0)}, s_{(1)}, \ldots, s_{(n)})$, given by:

$$\begin{aligned} s_{(0)} &= 0 \\ s_{(k)} &= \sum_{j=0}^{k-1} p_{(j)}\,, \; k = 1, 2, \ldots, n \end{aligned} \tag{5}$$

as a piecewise linear function of $\frac{k}{n}$ results in the family of Lorenz curves. A uniformly distributed sequence where $p_{(j)} = \frac{1}{n}$ produces the linear Lorenz curve represented by the solid line in Figure 1. With decreasing uniformity, the signal approaches a constant with probability sequence element $p_{(n-1)} = 1$, producing convex curves illustrated by the dashed lines of Figure 1. The LIM is defined by the area under the Lorenz curve above the $\frac{k}{n}$ axis [7-10]. This measure is expressed by:

$$\begin{aligned} \mu_{LIM} &= \tfrac{1}{n}(\tfrac{1}{2} + p_{(n-2)} + 2p_{(n-3)} + \cdots \\ &\quad + (n-1)p_{(0)}) \\ &= \tfrac{1}{n}(\tfrac{1}{2} + \sum_{j=0}^{n-1}(n-1-j)p_{(j)}) \end{aligned} \tag{6}$$

Accordingly, constant signals result in the minimum value $\mu_{LIM} = \frac{1}{2n}$, and uniformly distributed signals produce the maximum value $\mu_{LIM} = \frac{1}{2}$. Similarly, the MAE is expressed in terms of the unordered probability sequence p according to:

$$\begin{aligned} \mu_{MAE} &= p_1 + 2p_2 + \ldots + (n-1)p_{n-1} \\ &= \sum_{j=0}^{n-1} j p_j \end{aligned} \tag{7}$$

II. Methods

A theorem is presented providing a linear asymptotic bound of the LIM by the MAE/signal quantization number ratio for the difference signal case. This theory motivates performing simulations producing results which, although not directly obtainable from the theorem, illustrate the utility of the LIM. The simulations consist of employing 2-dimensional gaussian and magnetic resonance (MR) images as uncorrupted references, and corrupting these images with approximately zero mean gaussian, uniform, and combined gaussian and uniform additive noise with various standard deviations. Mean and median filtering are performed on these corrupted images in order to demonstrate the effect of standard filtering operations on the LIM in quantifying filtering performance compared to the entropy and minimum quantization number. The LIM, entropy, and minimum quantization number are computed for each image and plotted as a function of the MAE. Accordingly, the LIM is demonstrated to vary approximately linearly with the MAE for these simulations.

III. Results

A. Theory

The bound of the LIM presented below follows from a result in the rearrangement of finite sets of variables provided in reference [11] which is presented without proof as a lemma.

Lemma 1: Given a finite sequence with real valued, finite elements $a = (a_0, a_1, \ldots, a_{n-1})$ with the rearrangement in ascending order denoted by $a_{(j)} = (a_{(0)}, a_{(1)}, \ldots, a_{(n-1)})$ such that $a_{(0)} \leq a_{(1)} \leq, \ldots, \leq a_{(n-1)}$

Then:

$$\sum_{j=0}^{n-1}(n-1-j)a_{(j)} \leq \sum_{j=0}^{n-1} j a_j \tag{8}$$

In the general form of this lemma, the sum of the pairwise element products of two sequences is a minimum when the elements are monotonic in opposite senses and a maximum when monotonic in the same

sense. The LIM bound for the difference signal case is subsequently provided as a theorem.

Theorem 1: Given:

i) The finite sequences $a = (a_0, a_1, \ldots, a_{m-1})$ and $\hat{a} = (\hat{a}_0, \hat{a}_1, \ldots, \hat{a}_{m-1})$ with elements possessing at most n distinct finite integer values

ii) The probability sequence p derived from the histogram h of $|\hat{a} - a|$

iii) The Lorenz information measure μ_{LIM} and the mean absolute error μ_{MAE} defined by:

$$\mu_{LIM} = \frac{1}{n}\left(\frac{1}{2} + \sum_{j=0}^{n-1}(n-1-j)p_{(j)}\right)$$

$$\mu_{MAE} = \sum_{j=0}^{n-1} j p_j$$

Then:

$$\mu_{LIM} \leq \frac{1}{n}(\mu_{MAE} + \frac{1}{2}) \qquad (9)$$

Proof: Rearranging Equation (6) results in:

$$n\mu_{LIM} - \frac{1}{2} = \sum_{j=0}^{n-1}(n-1-j)p_{(j)} \qquad (10)$$

Since the sequences $((n-1),(n-2),\ldots,1,0)$ and $(p_{(0)}, p_{(1)}, \ldots, p_{(n-1)})$ are monotonic in opposite senses, applying Lemma 1 results in:

$$\sum_{j=0}^{n-1}(n-1-j)p_{(j)} \leq \sum_{j=0}^{n-1} j p_j \qquad (11)$$

Applying this inequality to Equation (10) and rearranging terms results in:

$$\mu_{LIM} \leq \frac{1}{n}(\mu_{MAE} + \frac{1}{2})$$

completing the proof. □

As a corollary, for increasingly large μ_{MAE} such that $\mu_{MAE} \gg \frac{1}{2}$, then $\mu_{LIM} \lesssim \frac{\mu_{MAE}}{n}$ where $\lesssim$ denotes asymptotically less than or equal to resulting from this μ_{MAE} approximating condition. Accordingly, the LIM is asymptotically bounded by the ratio of the MAE and the signal quantization number.

The above development considers probability sequences constructed from the difference signal $|\hat{a}-a|$. However, as discussed previously, the uncorrupted reference signal a is unknown in general, resulting in an undefined MAE. Alternatively, if the MAE is computed according to Equation (7) assuming that p is simply the corrupted signal rather than the difference signal probability sequence, the resulting measure is the absolute mean, or simply the mean, for strictly non-negative $\hat{a}$. Clearly, the mean is not a sensitive performance measure for signals with additive zero-mean noise processes. Conversely, the LIM is meaningfully defined for any discrete signal as a function of the probability sequence, independent of a reference signal, similar to H and 2^H. Of course, the domain (time/space) information of the MAE resulting from the difference signal structure of Equation (2) is lost in each of the other measures. Therefore, since the above theoretical result based on difference signals is suggestive of a relation to the MAE but not directly applicable to the general case where the reference signal does not exist, simulations are employed to demonstrate the utility of the LIM. Accordingly, for the simulations, the MAE is computed using the reference signal as required, while the LIM, H, and 2^H are evaluated from the corrupted signal only.

B. Simulations

Simulations are provided to demonstrate the use of the LIM in imaging applications. Uncorrupted and corrupted 8 bit, 64 × 64 gaussian and 256 × 256 MR images and the filtered results are generated. The LIM, minimum quantization number, and entropy are computed exclusively from the corrupted images, while the MAE is computed from the corrupted/uncorrupted difference images, as required by the definition. The uncorrupted image values are contained in the set $(64, 65, \ldots, 192)$. Six corrupted gaussian images are generated with gaussian additive noise standard deviations contained in (1.1,2.1,4.0,8.0,16.0,19.6). Similarly, the gaussian image is corrupted with uniform and combined gaussian and uniform noise with standard deviations (1.1,2.0,4.0,8.0,16.0,36.6) and (0.8,1.5,2.8,5.6,11.3,20.9), respectively. Subsequently, another 18 corrupted MR images are generated with gaussian, uniform, and combined noise standard deviations (1.0,2.0,4.0,8.0,16.0,17.5), (1.0,2.0,4.0,8.0,16.0,36.4), and (0.7,1.4,2.8,5.7,11.3, 20.2), respectively. Finally, mean and median filtering is performed and representative images are presented in Figures 2-3.

The computed MAE, LIM, and 2^H values for the unfiltered and filtered corrupted images are plotted in Figures 4-5. The LIM is demonstrated to provide slightly to significantly greater linearity with the MAE compared to 2^H, according to the correlation

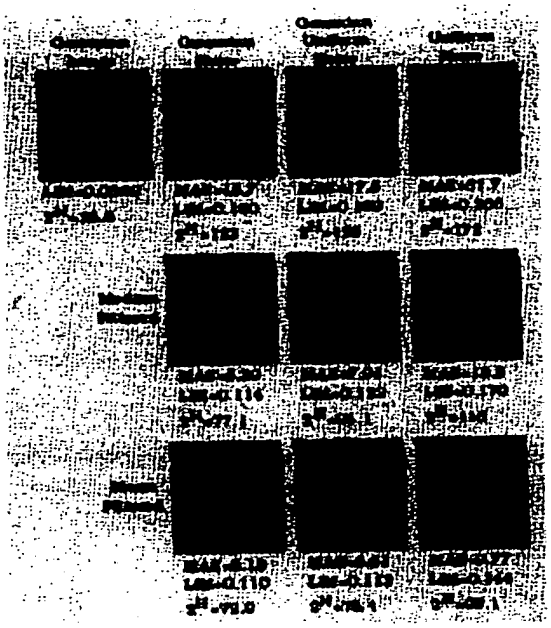


Figure 2. The uncorrupted and selected corrupted gaussian images and the median and mean filtered results are displayed from top to bottom respectively with the corresponding MAE, LIM, and minimum quantization number provided below.

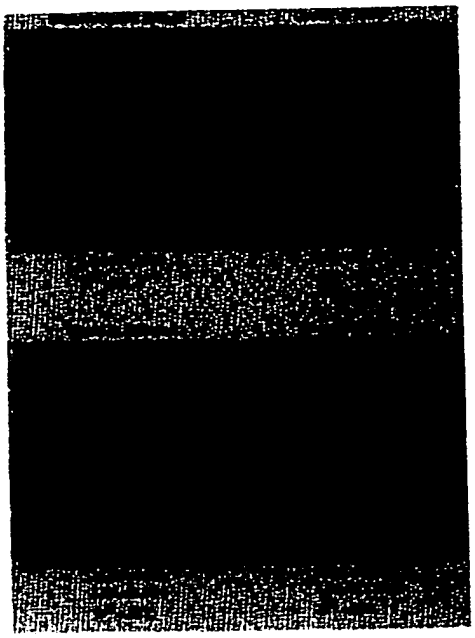


Figure 3. The uncorrupted and uniform noise corrupted MR images and the median and mean filtered results are displayed from top left to bottom right respectively with the corresponding MAE, LIM, and minimum quantization number provided below.

coefficient r^2. Indeed, the graphs of 2^H in Figures 4-5 appear to have a slightly logarithmic functional form close to the 2^H axis, similar to the obviously logarithmic functional form of the entropy graphs illustrated in Figure 6. The linear correlation coefficients of these entropy plots for the gaussian and MR images are 0.834 and 0.757, respectively: significantly less than the logarithmic correlation coefficients of 0.948 and 0.897, as presented in the figure. Thus, in these simulations, the LIM is demonstrated to be more linear with the MAE for the unfiltered and filtered corrupted images than the minimum quantization number and the obviously logarithmic entropy. This approximately linear relation is consistent with the theoretical results.

IV. Discussion

Statistical measures, such as the signal mean and standard deviation, are alternative quantities providing signal analysis capability. However, for the standard zero mean noise process commonly used in modeling, the mean is unchanged between uncorrupted and corrupted signals, producing no information about the benefits of signal processing algorithms performed on the corrupted signal. In addition, the standard deviation is observed also to exhibit less linearity with the MAE than the LIM in these simulations. As another alternative measure, the signal entropy is expected to be logarithmic in p by inspection. In order to suppress this logarithmic functional form, the minimum quantization number, representing the minimum amount of quantization levels required to represent the signal, is defined as 2^H. As a result, this measure displays some linearity with the MAE in the simulations, although less than the LIM. This is consistent with the asymptotic result of the corollary, where the MAE is much greater then $\frac{1}{2}$. Accordingly, the LIM provides a similar performance measure to the MAE in the analysis of filtering operations, for example, without requiring knowledge of an uncorrupted signal as a target for comparison of filtering results. Thus, general signal processing operations may be quantified relative to the LIM without the often nonexistent uncorrupted reference signals necessary for MAE computation.

Obviously, limitations exist in the interpretation

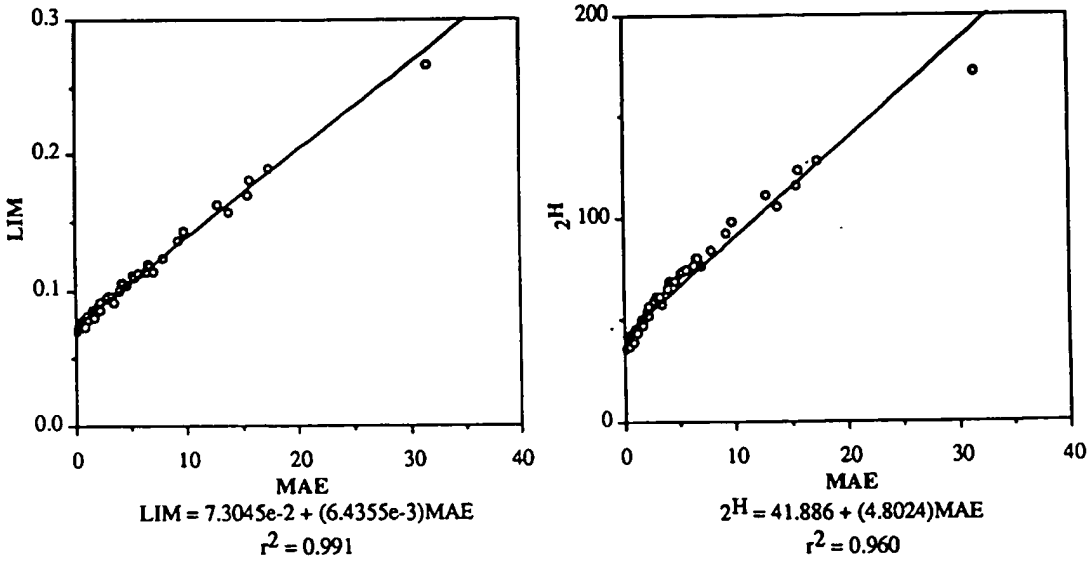


Figure 4. The LIM and minimum quantization number, from left to right, for the gaussian, uniform, and combined gaussian and uniform noise corrupted gaussian and the median and mean filtered images are plotted as a function of MAE with the linear regression equation and the correlation coefficient provided below.

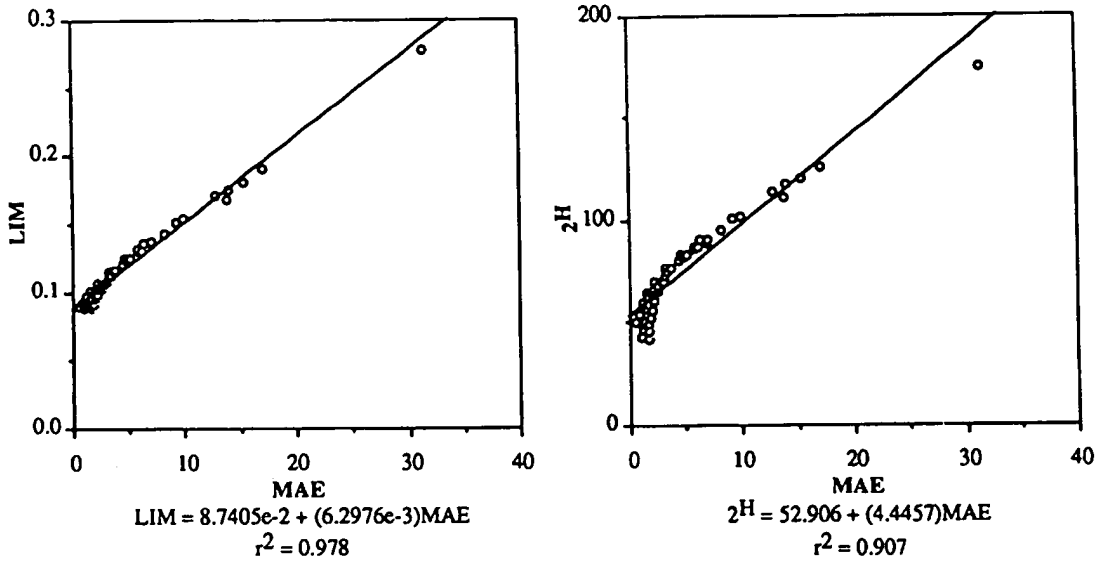


Figure 5. The LIM and minimum quantization number, from left to right, for the gaussian, uniform, and combined gaussian and uniform noise corrupted MR and the median and mean filtered images are plotted as a function of MAE with the linear regression equation and the correlation coefficient provided below.

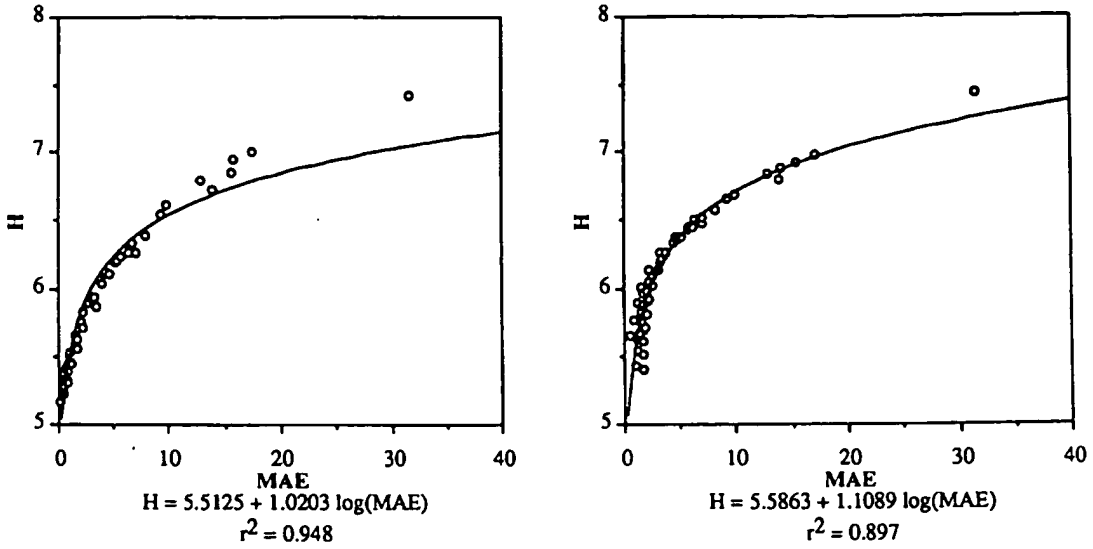


Figure 6. The entropy for the gaussian, uniform, and combined gaussian and uniform noise corrupted gaussian, on the left, and MR, on the right, and the median and mean filtered images are plotted as a function of MAE with the logarithmic regression equation and the correlation coefficient provided below.

of the LIM. For example, for signals which are approximately uniformly distributed over the quantization value set, the LIM approaches the maximum value of $\frac{1}{2}$, and addition of noise obviously increases the MAE without a similar increase in the LIM. This observation holds equally for both H and 2^H. Thus, the LIM, as well as H and 2^H, are more meaningful when applied to signals with significantly nonuniformly distributed histograms, which are more typically observed in actual signals acquired from real sources.

V. Conclusion

The LIM is demonstrated to be approximately linearly related to the MAE. The theoretical LIM bound by the MAE/signal quantization number ratio for the difference signal case motivates the simulations which illustrate this property for unfiltered and mean and median filtered gaussian, uniform, and combined noise corrupted gaussian and MR images. In this investigation, the LIM demonstrates greater linearity with the MAE than the entropy and the minimum quantization number. Thus, the LIM varys approximately linearly with the MAE providing a similar performance measure without requiring an uncorrupted reference signal for computation.

References

[1] A. N. Kolmogorov, S. V. Fomin, *Introductory Real Analysis*, Dover Publications, Inc., New York, NY, 1970.

[2] I. Daubechies, *Ten Lectures on Wavelets*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992.

[3] A. M. Pinkus, *On L^1-Approximation*, Cambridge University Press, Cambridge, UK, 1989.

[4] A. Torchinsky, *Real Variables*, Addison-Wesley Publishing Company, Redwood City, CA, 1988.

[5] Y. Dodge, Editor, *Statistical Data Analysis Based on the L_1-norm and Related Methods*, Elsevier Science Publishing Company, New York, NY, 1987.

[6] R. M. Capocelli, A. De Santis, I. J. Taneja, "Bounds on the Entropy Series", *IEEE Transactions on Information Theory*, Volume 34, Number 1, January, 1988, Pages 134-138.

[7] M. O. Lorenz, "Methods of Measuring the Concentration of Wealth", *Publications of the American Statistical Association*, Volume 9, 1905, Pages 209-219.

[8] A. W. Marshall, I. Olkin, *Inequalities: Theory of Majorization and Its Applications*, Academic Press, New York, NY, 1979.

[9] B. C. Arnold, *Majorization and the Lorenz Order: A Brief Introduction*, Springer-Verlag, Berlin, Germany, 1987.

[10] S. K. Chang, *Principles of Pictorial Information Systems Design*, Prentice Hall, Englewood Cliffs, NJ, 1989.

[11] G. H. Hardy, J. E. Littlewood, G. Polya, *Inequalities*, 2nd Edition, Cambridge University Press, London, England, 1959.

Histogram Refinement for Content-Based Image Retrieval

Greg Pass Ramin Zabih*

Computer Science Department
Cornell University
Ithaca, NY 14853
gregpass,rdz@cs.cornell.edu
http://www.cs.cornell.edu/home/rdz/refinement.html

Abstract

Color histograms are widely used for content-based image retrieval. Their advantages are efficiency, and insensitivity to small changes in camera viewpoint. However, a histogram is a coarse characterization of an image, and so images with very different appearances can have similar histograms. We describe a technique for comparing images called histogram refinement, *which imposes additional constraints on histogram based matching. Histogram refinement splits the pixels in a given bucket into several classes, based upon some local property. Within a given bucket, only pixels in the same class are compared. We describe a split histogram called a* color coherence vector *(CCV), which partitions each histogram bucket based on spatial coherence. CCV's can be computed at over 5 images per second on a standard workstation. A database with 15,000 images can be queried using CCV's in under 2 seconds. We demonstrate that histogram refinement can be used to distinguish images whose color histograms are indistinguishable.*

1 Introduction

Many applications require methods for comparing images based on their overall appearance. Color histograms are a popular solution to this problem, and are used in systems like QBIC [2] and Chabot [6]. Color histograms are computationally efficient, and generally insensitive to small changes in camera position. However, a color histogram provides only a very coarse characterization of an image; images with similar histograms can have dramatically different appearances. For example, the images shown in figure 1 have similar color histograms.

In this paper we describe a method which imposes additional constraints on histogram based matching. In *histogram refinement*, the pixels within a given bucket are split into classes based upon some local property. Split histograms are compared on a bucket by bucket basis, similar to standard histogram matching. Within a given bucket, only pixels with the same property are compared. Two images with identical color histograms can have different split histograms; thus, split histograms create a finer distinction than color histograms. This is particularly important for large image databases, in which many images can have similar color histograms.

Figure 1: Two images with similar color histograms

We have experimented with a split histogram called a *color coherence vector* (CCV), which partitions pixels based upon their spatial coherence. A coherent pixel is part of some sizable contiguous region, while an incoherent pixel is not. While the two images shown in figure 1 have similar color histograms, their CCV's are very different.[1] For example, red pixels appear in both images in similar quantities. In the left image the red pixels (from the flowers) are widely scattered, while in the right image the red pixels (from the golfer's shirt) form a single coherent region.

We begin with a review of color histograms. In section 3 we describe histogram refinement, and present two examples that capture spatial information. Section 4 provides examples of refinement-based image queries and shows that they can give superior results to color histograms. We compare our work with some recent algorithms [5, 8, 9, 10] that also combine spatial information with color histograms.

2 Color Histograms

Color histograms are frequently used to compare images. Examples of their use in multimedia applications include scene break detection and querying a database of images [7, 6, 2]. Color histograms are popular because they are trivial to compute, and tend to

*To whom correspondence should be addressed

[1] The color images used in this paper can be found at http://www.cs.cornell.edu/home/rdz/refinement.html.

0-8186-7620-5/96 $5.00 © 1996 IEEE

be robust against small changes in camera viewpoint. For example, Swain and Ballard [12] describe the use of color histograms for identifying objects. Stricker and Swain [11] analyze the information capacity of color histograms.

We will assume that all images are scaled to contain the same number of pixels M. We discretize the colorspace of the image such that there are n distinct (discretized) colors. A color histogram H is a vector $\langle h_1, h_2, \ldots, h_n \rangle$, in which each bucket h_j contains the number of pixels of color j in the image. Typically images are represented in the RGB colorspace, with a few of the most significant bits per color channel.

For a given image I, the color histogram H_I is a compact summary of the image. A database of images can be queried to find the most similar image to I, and can return the image I' with the most similar color histogram $H_{I'}$. Color histograms are typically compared using the L_1-distance or the L_2-distance, although more complex measures have also been considered [4].

3 Histogram Refinement

In *histogram refinement* the pixels of a given bucket are subdivided into classes based on local features. There are many possible features, including texture, orientation, distance from the nearest edge, relative brightness, etc. Histogram refinement prevents pixels in the same bucket from matching each other if they do not fall into the same class. Pixels in the same class can be compared using any standard method for comparing histogram buckets (such as the L_1 distance). This allows fine distinctions that cannot be made with color histograms.

As a simple example of histogram refinement, consider a positional refinement where each pixel in a given color bucket is classified as either "in the center" of the image, or not. Specifically, the centermost 75% of the pixels are defined as the "center". This produces a split histogram in which the pixels of color buckets are loosely constrained by their location in the image. The resulting split histograms can be compared using the L_1 distance. We will call this simple form of histogram refinement *centering refinement*.

Color coherence vectors

CCV's are a more sophisticated form of histogram refinement, in which histogram buckets are partitioned based on spatial coherence. Our coherence measure classifies pixels as either coherent or incoherent. A coherent pixel is a part of a sizable contiguous region, while an incoherent pixel is not. A *color coherence vector* represents this classification for each color in the image.

The initial stage in computing a CCV is similar to the computation of a color histogram. We first blur the image slightly by replacing pixel values with the average value in a small local neighborhood (currently including the 8 adjacent pixels). We then discretize the colorspace, such that there are only n distinct colors in the image.

The next step is to classify the pixels within a given color bucket as either coherent or incoherent. A coherent pixel is part of a large group of pixels of the same color, while an incoherent pixel is not. We determine the pixel groups by computing connected components. A connected component C is a maximal set of pixels such that for any two pixels $p, p' \in C$, there is a path in C between p and p'. We compute connected components using 4-connected neighbors within a given discretized color bucket. We classify pixels as either coherent or incoherent depending on the size in pixels of its connected component. A pixel is coherent if the size of its connected component exceeds a fixed value τ; otherwise, the pixel is incoherent.

For a given discretized color, some of the pixels with that color will be coherent and some will be incoherent. Let us call the number of coherent pixels of the j'th discretized color α_j and the number of incoherent pixels β_j. Clearly, the total number of pixels with that color is $\alpha_j + \beta_j$, and so a color histogram would summarize an image as $\langle \alpha_1 + \beta_1, \ldots, \alpha_n + \beta_n \rangle$. Instead, for each color we compute the pair (α_j, β_j) which we will call the *coherence pair* for the j'th color. The *color coherence vector* for the image consists of $\langle (\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n) \rangle$. This is a vector of coherence pairs, one for each discretized color.

In our experiments, all images were scaled to contain $M = 38,976$ pixels, and we have used $\tau = 300$ pixels (so a region is classified as coherent if its area is about 1% of the image). With this value of τ, an average image in our database consists of approximately 75% coherent pixels, with a standard deviation of 11%.

Two images I, I' can be compared using their CCV's, for example by using the L_1 distance. Let the coherence pairs for the j'th color bucket be (α_j, β_j) in I and (α'_j, β'_j) in I'. Using the L_1 distance to compare CCV's, the j'th bucket's contribution to the distance between I and I' is

$$\Delta_{CCV} = |(\alpha_j - \alpha'_j)| + |(\beta_j - \beta'_j)|. \quad (1)$$

Note that when using color histograms to compare I and I', the j'th bucket's contribution is

$$\Delta_{CH} = |(\alpha_j + \beta_j) - (\alpha'_j + \beta'_j)|. \quad (2)$$

It follows that CCV's create a finer distinction than color histograms. A given color bucket j can contain the same number of pixels in I as in I', but these pixels may be entirely incoherent in I and entirely coherent in I' (i.e., $\alpha = \beta' = 0$). Formally, $\Delta_{CH} \leq \Delta_{CCV}$ follows from equations 1 and 2, and the fact that the L_1 distance obeys the triangle inequality.

4 Experimental Results

We have implemented histogram refinement, and have used it for image retrieval from a large database. Our database consists of 14,554 images, which are drawn from a variety of sources. Our largest sources include the 11,667 images used in Chabot [6], the 1,440 images used in QBIC [2], and a 1,005 image database available from Corel. In addition, we included a few groups of images in PhotoCD format. Finally, we have taken a number of MPEG videos from the Web and segmented them into scenes. We have added one or

two images from each scene to the database, totaling 349 images. The image database thus contains a wide variety of imagery.

We have compared our results with a number of color histogram variants. These include the L_1 and L_2 distances, with both 64 and 512 color buckets. We include a small amount of smoothing as it empirically improved performance. On our database, the L_1 distance with the 64-bucket RGB colorspace gave the best results, and is used as a benchmark.

Hand examination of our database revealed 75 pairs of images which contain different views of the same scene. Examples are shown in figures 2 and 3. One image is selected as a query image, and the other represents a "correct" answer. In each case, we have shown where the second image ranks, when similarity is computed using color histograms or using histogram refinement. Specifically, results are shown using CCV's, centering refinement, and a successive refinement technique described in section 6.1. The color images shown are available at http://www.cs.cornell.edu/home/rdz/refinement.html.

4.1 Centering refinement results

In 69 of the 75 cases, centering refinement produced better results, while in 4 cases it produced worse results (there were 2 cases where the ranks did not change). The average change in rank due to centering refinement was an improvement of 55 positions (this included all 75 cases). The average percentage change in rank was an improvement of 41%. In the 69 cases where centering refinement performed better than color histograms, the average improvement in rank was 60 positions, and the average percentage improvement was 49%. In the 4 cases where color histograms performed better than centering refinement, the average rank improvement was 10 positions. We have not yet analyzed these 4 cases to determine why centering refinement fails.

To analyze the statistical significance of this data, we formulate the null hypothesis H_0 which states that centering refinement is equally likely to cause a positive change in ranks (i.e., an improvement) or a negative change. We will discard the 2 ties to simplify the analysis. Under H_0, the expected number of positive changes is 36.5, with a standard deviation of $\sqrt{73}/2 \approx 4.27$. The actual number of positive changes is 69, which is more than 7.6 standard deviations greater than the number expected under H_0. We can therefore reject H_0 at any standard significance level (such as 99.9%).

4.2 CCV results

In 68 of the 75 cases, CCV's produced better results, while in 7 cases they produced worse results. The average change in rank due to CCV's was an improvement of 68 positions (note that this included the 7 cases where CCV's do worse). The average percentage change in rank was an improvement of 35%. In the 68 cases where CCV's performed better than color histograms, the average improvement in rank was 77 positions, and the average percentage improvement was 56%. In the 7 cases where color histograms performed better, the average improvement was 17 positions.

The null hypothesis H_0 states that CCV's are equally likely to cause a positive change in ranks (i.e., an improvement) or a negative change. Under H_0, the expected number of positive changes is 37.5, with a standard deviation of $\sqrt{75}/2 \approx 4.33$. The actual number of positive changes is 68, which is more than 7 standard deviations greater than the number expected under H_0. We can therefore reject H_0 at any standard significance level (such as 99.9%).

When CCV's produced worse results, it was always due to a change in overall image brightness (i.e., the two images were almost identical, except that one was brighter than the other). Because CCV's use discretized color buckets for segmentation, they are more sensitive to changes in overall image brightness than color histograms. We believe that this difficulty can be overcome by using a better colorspace than RGB, as we discuss in section 6.2.

4.3 Efficiency

We have experimented with a number of different techniques for histogram refinement. CCV's are the most computationally expensive method of these, and will be our focus in discussing efficiency.

There are two phases to the computation involved in querying an image database. First, when an image is inserted into the database, a CCV must be computed. Second, when the database is queried, some number of the most similar images must be retrieved. Most methods for content-based indexing include these distinct phases. For both color histograms and CCV's, these phases can be implemented in linear time with a single pass over the image.

We ran our experiments on a 50 MHz SPARCstation 20, and provide the results from color histogramming for comparison. Color histograms can be computed at 67 images per second, while CCV's can be computed at 5 images per second. Using color histograms, 21,940 comparisons can be performed per second, while with CCV's 7,746 can be performed per second. The images used for benchmarking are 232×168. Both implementations are preliminary, and the performance can definitely be improved.

5 Related Work

Our work has focused on the use of spatial information to refine color histograms. Recently, several authors have proposed algorithms for comparing images that combine spatial information with color histograms. Hsu *et al.* [5] attempts to capture the spatial arrangement of the different colors in the image, in order to perform more accurate content-based image retrieval. Rickman and Stonham [8] randomly sample the endpoints of small triangles and compare the distributions of these triplets. Smith and Chang [9] concentrate on queries that combine spatial information with color. Stricker and Dimai [10] divide the image into five partially overlapping regions and compute the first three moments of the color distributions in each image. We will discuss each approach in turn.

Hsu [5] begins by selecting a set of representative colors from the image. Next, the image is partitioned

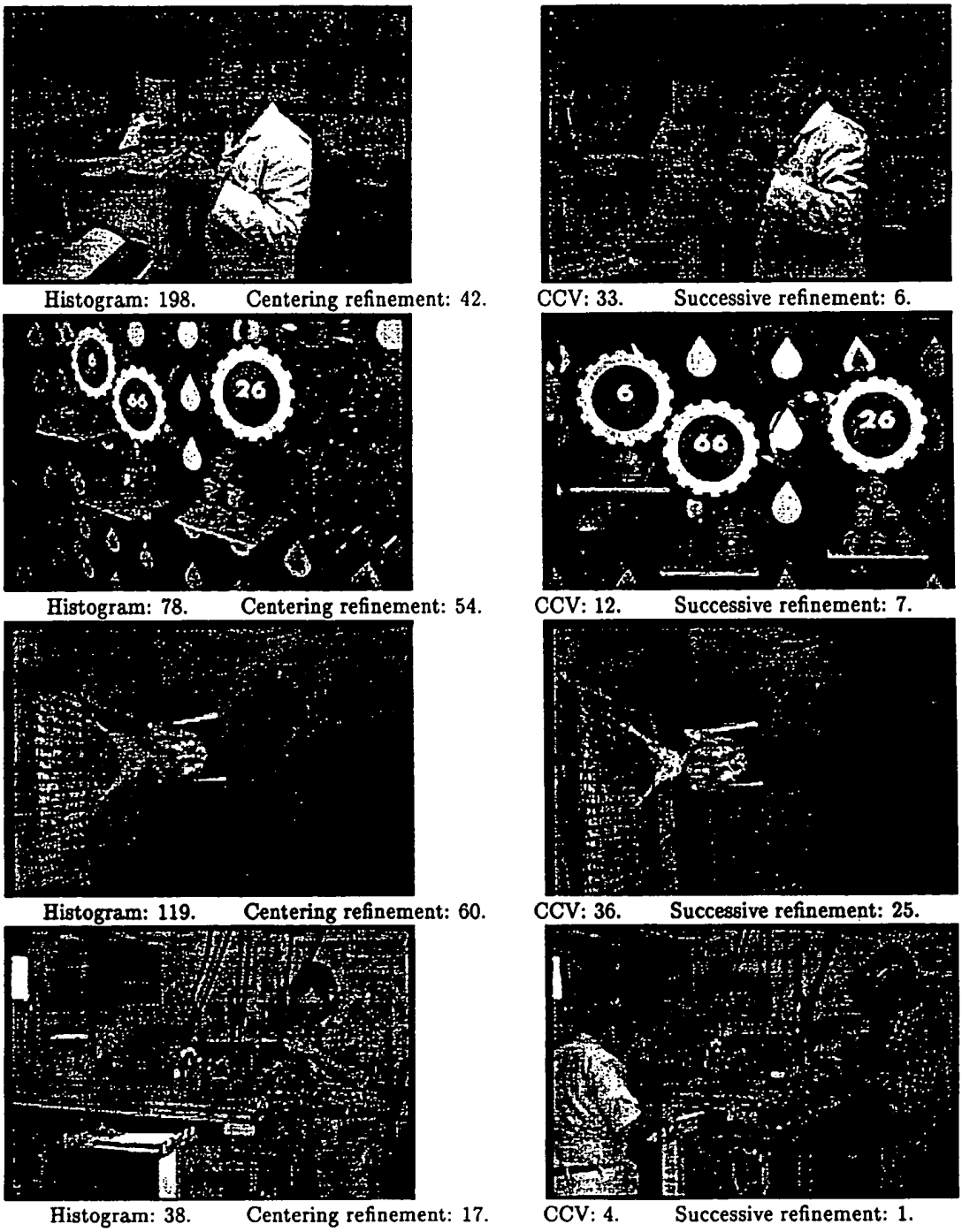
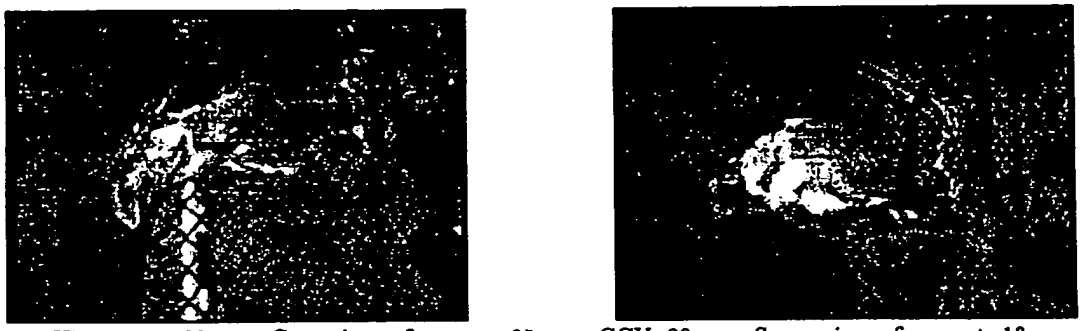
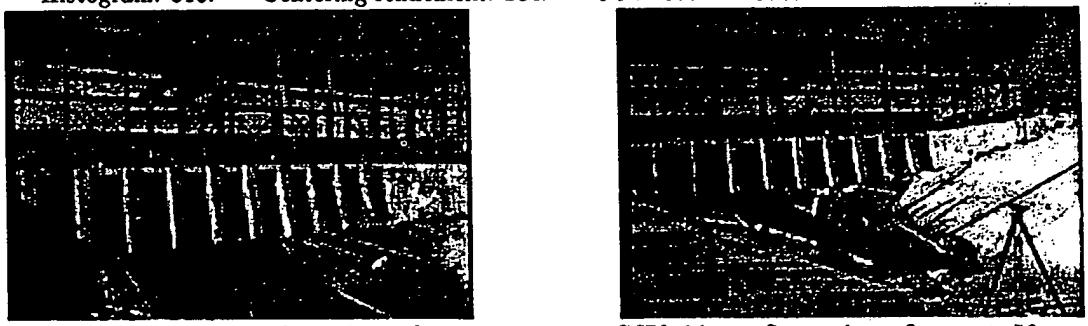


Figure 2: Example queries with their partner images, plus ranks under various methods. Lower ranks indicate better performance.

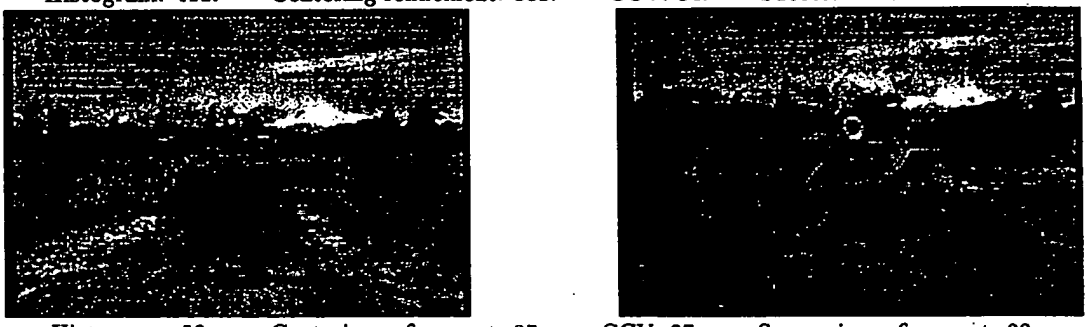


Histogram: 88. Centering refinement: 35. CCV: 20. Successive refinement: 13.

Histogram: 310. Centering refinement: 214. CCV: 177. Successive refinement: 160.



Histogram: 411. Centering refinement: 282. CCV: 84. Successive refinement: 56.



Histogram: 50. Centering refinement: 37. CCV: 27. Successive refinement: 22.

Figure 3: Additional example queries with ranks. Lower ranks indicate better performance.

into rectangular regions, where each region is predominantly a single color. The partitioning algorithm makes use of maximum entropy. The similarity between two images is the degree of overlap between regions of the same color. Hsu presents results from a database with 260 images, which show that their approach can give better results than color histograms.

While the authors do not report running times, it appears that Hsu's method requires substantially more computation than the approach we describe. A CCV can be computed in a single pass over the image, with a small number of operations per pixel. Hsu's partitioning algorithm in particular appears much more computationally intensive than our method. Hsu's approach can be extended to be independent of orientation and position, but the computation involved is quite substantial. In contrast, our method is naturally invariant to orientation and position.

Rickman and Stonham [8] randomly sample pixel triples arranged in an equilateral triangle with a fixed side length. They use 16 levels of color hue, with non-uniform quantization. Approximately a quarter of the pixels are selected for sampling, and their method stores 372 bits per image. They report results from a database of 100 images.

Smith and Chang's algorithm also partitions the image into regions, but their approach is more elaborate than Hsu's. They allow a region to contain multiple different colors, and permit a given pixel to belong to several different regions. Their computation makes use of histogram back-projection [12] to back-project sets of colors onto the image. They then identify color sets with large connected components.

Smith and Chang's image database contains 3,100 images. Again, running times are not reported, although their algorithm does speed up back-projection queries by pre-computing the back-projections of certain color sets. Their algorithm can also handle certain kinds of queries that our work does not address; for example, they can find all the images where the sun is setting in the upper left part of the image.

Stricker and Dimai [10] compute moments for each channel in the HSV colorspace, where pixels close to the border have less weight. They store 45 floating point numbers per image. Their distance measure for two regions is a weighted sum of the differences in each of the three moments. The distance measure for a pair of images is the sum of the distance between the center regions, plus (for each of the 4 side regions) the minimum distance of that region to the corresponding region in the other image, when rotated by 0, 90, 180 or 270 degrees. Because the regions overlap, their method is insensitive to small rotations or translations. Because they explicitly handle rotations of 0, 90, 180 or 270 degrees, their method is not affected by these particular rotations. Their database contains over 11,000 images, but the performance of their method is only illustrated on 3 example queries. Like Smith and Chang, their method is designed to handle certain kinds of more complex queries that we do not consider.

6 Extensions

There are a number of ways in which our histogram refinement could be extended and improved. One generalization is to further subdivide split histograms based on additional features; we refer to this process as *successive refinement*. Another extension centers on improving the choice of colorspace.

6.1 Successive refinement

In *successive refinement* the buckets in a split histogram are further subdivided based on additional features. Much as we distinguish between pixels of similar color by coherence, we can distinguish between pixels of similar coherence by some additional feature. We can apply this method repeatedly; each refinement imposes an additional constraint on what it means for two pixels to be similar.

We have implemented a simple successively refined histogram. A color histogram was first split with coherence constraints (to create a CCV). Successive refinement was enforced on both the coherent and incoherent pixels of the CCV. We used the centering refinement introduced in section 3. With successive refinement, pixels are divided into four classes based on coherence versus incoherence, and on whether or not they were in the centermost 75% of the image. The L_1 distance was used as a comparison measure. Examples of the successively refined histogram's performance are shown in figures 2 and 3. These preliminary results seem promising.

We have also investigated successive refinement based on intensity gradients. Again, the initial refinement was based on coherence, and the successive refinement was enforced identically on coherent and incoherent pixels. We have further classified pixels based on the gradient magnitude or on the gradient direction. The results we obtained are quite preliminary, but they seem to indicate a statistically significant improvement over CCV's.

The best system of constraints to impose on the image is an open issue. Any combination of features might give effective results, and there are many possible features to choose from. However, it is possible to take advantage of the temporal structure of a successively refined histogram. One feature might serve as a filter for another feature, by ensuring that the second feature is only computed on pixels which already possess the first feature.

For example, the perimeter-to-area ratio can be used to classify the relative shapes of color regions. If we used this ratio as an initial refinement on color histograms, incoherent pixels would result in statistical outliers, and thus give questionable results. This feature is better employed after the coherent pixels have been segregated. Refining a histogram not only makes finer distinctions between pixels, but functions as a statistical filter for successive refinements.

6.2 Choice of colorspace

Many researchers spend considerable effort on selecting a good set of colors. Hsu [5], for example, assumes that the colors in the center of the image are more important than those at the periphery, while Smith and Chang [9] use several different thresholds to

extract colors and regions. A wide variety of different colorspaces have also been investigated for content-based image retrieval, such as the opponent-axis colorspace [12] and the Munsell colorspace [2].

The choice of colorspace is a particularly significant issue for CCV's, since they use the discretized color buckets to segment the image. A perceptually uniform colorspace, such as CIE Lab, should result in better segmentations and improve the performance of CCV's. A related issue is the color constancy problem, which causes objects of the same color to appear rather differently depending upon the lighting conditions. The simplest effect of color constancy is a change in overall image brightness; this is responsible for the negative examples obtained in our experiments with CCV's. Standard histogramming methods are sensitive to image gain. More sophisticated methods, such as color ratio histograms [3] or the use of color moments [10], might alleviate this problem. These methods, like most proposed improvements to color histograms, can also be used in histogram refinement. For example, color moments could be computed separately for coherent and incoherent pixels.

7 Conclusions

We have described a method for imposing additional constraints on histogram based matching called histogram refinement. This idea can be extended by placing further constraints on the split histogram itself. Both histogram refinement and successive refinement are general methods for improving the performance of histogram based matching. If the initial histogram is a color histogram, and it is refined based on coherence, then the resulting split histogram is a CCV. But there is no requirement that this refinement be based on coherence, or even that the initial histogram be based on color.

Most research in content-based image retrieval has focused on query by example (where the system automatically finds images similar to an input image). However, other types of queries are also important. For example, it is often useful to search for images in which a subset of another image (e.g. a particular object) appears. This would be particularly useful for queries on a database of videos. One approach to this problem might be to generalize histogram backprojection [12] to separate pixels based on spatial coherence, or some other local property.

It is clear that larger and larger image databases will demand more complex similarity measures. This added time complexity can be offset by using efficient, coarse measures that prune the search space by removing images which are clearly not the desired answer. Measures which are less efficient but more effective can then be applied to the remaining images. Baker and Nayar [1] have begun to investigate similar ideas for pattern recognition problems. To effectively handle large image databases will require a balance between increasingly fine measures (such as histogram refinement) and efficient coarse measures.

Acknowledgments

We wish to thank Virginia Ogle for giving us access to the Chabot imagery, and Thorsten von Eicken for supplying additional images. Greg Pass has been supported by Cornell's Alumni-Sponsored Undergraduate Research Program. We also thank Vera Kettnaker and Justin Miller for helping produce the data.

References

[1] Simon Baker and Shree Nayar. Pattern rejection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 544–549, 1996.

[2] M. Flickner *et al.* Query by image and video content: The QBIC system. *IEEE Computer*, 28(9):23–32, September 1995.

[3] Brian V. Funt and Graham D. Finlayson. Color constant color indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):522–529, May 1995.

[4] J. Hafner, H. Sawhney, W. Equitz, M. Flickner, and W. Niblack. Efficient color histogram indexing for quadratic form distance functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(7):729–736, July 1995.

[5] Wynne Hsu, T. S. Chua, and H. K. Pung. An integrated color-spatial approach to content-based image retrieval. In *ACM Multimedia Conference*, pages 305–313, 1995.

[6] Virginia Ogle and Michael Stonebraker. Chabot: Retrieval from a relational database of images. *IEEE Computer*, 28(9):40–48, September 1995.

[7] Alex Pentland, Rosalind Picard, and Stan Sclaroff. Photobook: Content-based manipulation of image databases. *International Journal of Computer Vision*, 18(3):233–254, June 1996.

[8] Rick Rickman and John Stonham. Content-based image retrieval using color tuple histograms. *SPIE proceedings*, 2670:2–7, February 1996.

[9] John Smith and Shih-Fu Chang. Tools and techniques for color image retrieval. *SPIE proceedings*, 2670:1630–1639, February 1996.

[10] Markus Stricker and Alexander Dimai. Color indexing with weak spatial constraints. *SPIE proceedings*, 2670:29–40, February 1996.

[11] Markus Stricker and Michael Swain. The capacity of color histogram indexing. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 704–708, 1994.

[12] Michael Swain and Dana Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

finding or retrieving websites only

Search and Ranking Algorithms for Locating Resources on the World Wide Web

Budi Yuwono

Department of Computer and Information Science
The Ohio State University
Columbus, Ohio, U.S.A.
yuwono-b@cis.ohio-state.edu

Dik L. Lee

Department of Computer Science
Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong
dlee@cs.ust.hk

Abstract

Applying information retrieval techniques to the World Wide Web (WWW) environment is a unique challenge, mostly because of its hypertext/hypermedia nature and the richness of the meta-information it provides. We present four keyword-based search and ranking algorithms for locating relevant WWW pages with respect to user queries. The first algorithm, Boolean Spreading Activation, extends the notion of word occurrence in Boolean retrieval model by propagating the occurrence of a query word in a page to other pages linked to it. The second algorithm, Most-cited, uses the number of citing hyperlinks between potentially relevant WWW pages to increase the relevance scores of the referenced pages over the referencing pages. The third algorithm, TFxIDF vector space model, is based on word distribution statistics. The last algorithm, Vector Spreading Activation, combines TFxIDF with the spreading activation model. We conducted an experiment to evaluate the retrieval effectiveness of these algorithms. From the results of the experiment, we draw conclusions regarding the nature of the WWW environment with respect to document ranking strategies.

1 Introduction

The World Wide Web (WWW) [4] has become one of the fastest growing applications on the Internet today. Its popularity can be attributed mainly to its uniform access method for various network information services and its hypermedia support which links a wide range of multimedia data physically distributed all around the world into a single gigantic virtual database. WWW also provides a powerful and easy to setup medium for almost any user on the Internet to disseminate information. More and more information has become available online through WWW, from personal data to scientific reports to up-to-the-minute satellite images. This information explosion leads to a problem commonly known as resource discovery problem [14]. In order to find interesting WWW pages, a user has to browse through many WWW sites. This is a very time consuming process.

Methods to relief the users from this information overflow problem have been explored by others, from creating a special Usenet [8] newsgroup[1] for announcing new WWW sites, to sharing personal hotlists (accessible from the owner's home pages), compiled lists and catalogs, to searchable full-text index databases. In this paper, we present a WWW index server designed to help users locate WWW pages using keyword search. Based on how the index is built, there are two categories of WWW searchable index servers, namely manually generated index servers and robot generated index servers.

Among the well known manually built index servers are Yahoo[2] and EInet Galaxy.[3] The main advantage of manual indexing is that Web pages can be organized, hierarchically or otherwise, by subject, such as the subject tree structure in Yahoo and EInet Galaxy. Of course, such categorizations are subjective and may be biased to the maintainer's knowledge and background. A slightly different scheme of manually generated index system is the one used by the Global Online Directory (GOLD[4]), among others, which allows any user to add an entry (a pointer to a Web page along with other information) into the index database. Similar to the above scheme is that of Archie-like Web server (ALIWEB[5]) [7]. Instead of users registering their WWW pages, ALIWEB retrieves index data from each of the participating WWW servers. This index data is prepared manually by the respective WWW server maintainers in a standard text format containing the description of information provided by the servers.

Our index server falls into the category of robot-generated index servers. Robot-based indexing is faster and more comprehensive than manual index-

[1] *comp.infosystems.www.announce* newsgroup.
[2] ⟨http://www.yahoo.com/⟩.
[3] ⟨http://galaxy.einet.net/⟩.
[4] ⟨http://www.gold.net/gold/⟩.
[5] ⟨http://web.nexor.co.uk/public/aliweb/aliweb.html⟩.

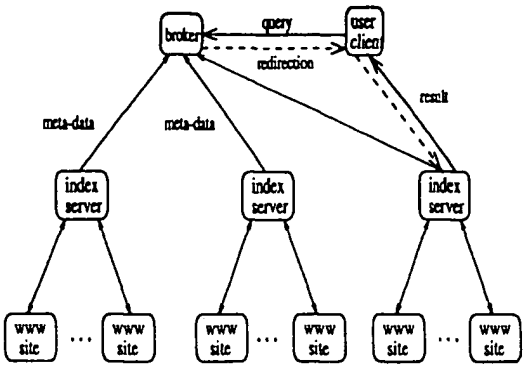


Figure 1: Multiserver organization and query routing.

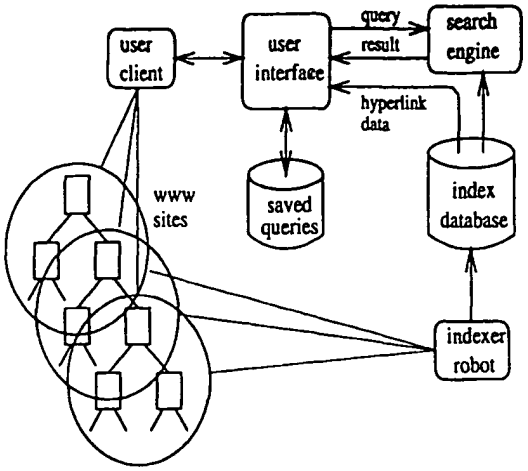


Figure 2: The WWW index server architecture.

ing and, since it is automated, it is easy to update the index as often as necessary. We discuss some of the robot-based index servers on the Internet in the last section of this paper.

Our WWW index server[6] takes a query from a user and returns a list of URL's (Uniform Resource Locators [1] or WWW page addresses) along with their titles, ranked by relevance score. Hyphens may be used to specify phrases so that the search algorithm only searches for occurrences of words in the same word ordering as they are in the phrase. For example, the query "computer-science" matches only pages containing the word "computer" immediately followed by the word "science". Our index server also allows a user to save a query, along with an optional single-line comment, on the server so that other users can share his/her discovery. Saved queries are stored in a list of clickable query statements. By clicking on one of these statements, a user can resubmit the query to the index server.

The index server is a key component of the Distributed World Wide Web Index and Search Engine (D-WISE) project, which is being conducted at the Hong Kong University of Science and Technology. Figure 1 illustrates the global architecture of D-WISE, where an index server covers a number of WWW sites belonging to a group based on geographical location, institution or other categories. For instance, the index server currently operational covers most of the sites in Hong Kong; similar index servers may be developed for a particular institution (e.g., covering all of the NASA sites). Each of such servers reports to one or more special servers, called the brokers. A broker maintains a catalog of meta-information which describes the topics covered by each of the index servers. A user client can send a query to one of the brokers. The broker then redirects the query to the index server which can potentially provide the best answer. This approach is similar to that of other server indexing methods such as GLOSS [6]. The emphasis of D-WISE is more on the schemes for meta-data exchange between index servers and brokers, and for automatic index server categorization. The detail of D-WISE is beyond the scope of this paper. In this paper, we describe the design and implementation of WISE, the index server, as a stand-alone system.

It is worth noting that our goal is not to replace browsing with keyword search, but to supplement it. We believe that browsing is an intuitive and appealing paradigm for accessing information. However, the search paradigm can bring the user closer to potentially relevant sites or pages quickly, from which browsing can be used to further explore interesting sites or pages in the neighborhood. The objective of this paper is to explore and evaluate several different search strategies, including new ones which are designed to take advantage of hyperlink and other meta-information specific to the WWW environment.

The rest of the report is organized as follows. In section 2, we describe the design of our index server in general. Section 3 presents the search algorithms in detail. Section 4 discusses the experiment we conducted to evaluate the search algorithms. Finally, in section 5 we discuss the conclusions drawn from the results of the experiment, plans for future study, and some comparisons with other WWW index servers.

2 System Description

Our WWW index server consists of two main components: an indexer robot and a search engine. Figure 2 illustrates the system architecture.

2.1 Indexer Robot

The indexer robot is an autonomous WWW browser which communicates with WWW servers using HTTP (Hypertext Transfer Protocol [2]). It visits

[6] The WWW index server currently covers most WWW servers in Hong Kong, and is publicly accessible at (http://www.cs.ust.hk/cgi-bin/IndexServer).

a given WWW site, traverses hyperlinks in a breadth-first manner, retrieves WWW pages, extracts keywords and hyperlink data from the pages, and inserts the keywords and hyperlink data into an index.

The index consists of a page-ID table, a keyword-ID table, a page-title table, a page modification-date table, a hyperlink table, and two index tables, namely, an inverted index and a forward index. The page-ID table maps each URL to a unique page-ID. The keyword-ID table maps each keyword to a unique keyword-ID. The page-title table maps every page-ID to the page's title. The page modification-date table maps every page-ID to the date when the page was last visited by the indexer robot. The hyperlink table maps each page-ID with two arrays of page-ID's, one array representing a list of pages referencing the page (incoming hyperlinks) and the other array representing a list of pages referenced by the page (outgoing hyperlinks). The inverted index table maps each keyword-ID to an array of (page-ID, word-position) pairs, each of which represents a page containing the keyword and the position of the word (order number) in the page. This word-position information is used in phrase searches mentioned in the introduction. Such information may also be useful for search strategies which take into account the distances between keywords. In this study, we do not investigate such strategies. The forward-index table maps a page-ID to an array of keyword-ID's representing keywords contained in the page. To obtain a fast access speed, hashing method is used to index each of these tables on the page-ID or keyword-ID attribute. The decision to store the index in separate tables instead of two large tables, one indexed by page-ID and the other indexed by keyword-ID, was based on its ease of maintenance and modularity. Moreover, the objective of this project is to develop algorithms that would work best in the WWW environment in terms of precision and recall. Thus, efficiency is not a primary concern at this stage of the project.

In extracting the keywords, we exclude high-frequency function words (stop-words), numbers, computer specific identifiers such as file-names, file directory paths, email addresses, network host names, and HTML (Hypertext Markup Language [3]) tags. To reduce storage overhead, the indexer robot only indexes words enclosed by HTML tags indicating tokens such as page titles, headings, hyperlink anchor names, words in bold-face, words in italic, and words in the first sentence of every list item. We assume that a WWW author will use these tags only on important sentences or words in his/her WWW pages. Thus, these words make good page identifiers. This is one of the advantages of adopting SGML (Standard General Markup Language), of which HTML is a subset. Of course, there may be other important words which are not enclosed by any of the above HTML tags. Words chosen as keywords are then stemmed by removing their suffices.

Resources using protocols other than HTTP (FTP, Gopher, Telnet, etc.) or in formats other than HTML text file (non-inline image, sound, video, binary, and other text files), click-able image maps, and CGI scripts are indexed by the anchor texts referencing them.

Periodic maintenance of the index files is performed bi-weekly by the indexer robot. First, the indexer robot checks the validity of every URL entry in the database by sending a special request to the WWW server containing the page to check whether the page has been modified since the time it was last accessed by the indexer robot. This special request, known as HEAD request, is a feature supported by HTTP. Non-routine index maintenance is also supported. This is performed at night in response to user requests received during the day to index new pages (URL's) or re-index updated pages. Such user requests are facilitated by an electronic form provided by the index server.

Our indexer robot has the capability of detecting looping paths, e.g., those caused by Unix symbolic links, and does not visit the same page more than once unless the page has been modified since the time it was last visited by the robot. The latter is made possible by supplying the last access date and time into the HTTP request.[7] As specified in the HTTP specification [2], the remote WWW server will not send the page content in response to the request if the page has not been modified since the specified time. Furthermore, the robot will not even send an HTTP request if the page was last accessed within the last 24 hours. This is to prevent the robot from sending more than one HTTP requests for the same page during a maintenance batch. To prevent the robot from endlessly roaming around from one server to the next, the robot accesses page at one site at a time and only references within the same site domain as that of the referencing page are traversed. Finally, the robot supports the proposed standard for robot exclusion[8] which prevents the robot from accessing places where, for various reasons, it is not welcome.

The indexer robot and the WWW robot is written in the C language. All index files are implemented using the GNU GDBM Database Manager library package [9].

2.2 Search Engine

The user interface to the search engine is a HTML form which can be invoked by standard WWW clients such as Mosaic and Netscape. The user types in the keywords and clicks on a submit button to send the query to the search engine.

Upon receiving a query, the search engine executes one of the ranking algorithms on the index database and produces a ranked list of URL's, from which the user can access the physical pages. Since the index database contains all of the information needed for ranking, the ranking process does not have to access to any WWW pages physically. If desired, the user

[7] Using the *If-Modified-Since* request header field.

[8] (http://info.webcrawler.com/mak/projects/robots/norobots.html).

can specify the maximum number of URL's to return and the ranking algorithm to use, instead of using the default setting. We discuss the ranking algorithms in detail in the next section.

It is clear that it is infeasible to search the WWW pages directly to compute the relevance scores without the help of the index. However, maintaining the currency of the index is a problem. We consider this a necessary tradeoff between speed and timeliness of the results. An immediate solution is to increase the frequency of index rebuild (notice that only the part of the index updated needs to be rebuilt). We are investigating within the D-WISE project efficient ways of organizing the index to facilitate detection of updates and reorganization in a WWW server.

The search engine and its gateway mechanism run as CGI scripts (external programs executable by a WWW server on behalf of WWW clients using a standard mechanism called Common Gateway Interface). These scripts are written in C code. Our current server is running under NCSA HTTPD version 1.3 WWW server.

3 Ranking Algorithms

In this paper, we explore four ranking algorithms, namely, (1) Boolean Spreading Activation, (2) Most-cited, (3) TFxIDF, and (4) Vector Spreading Activation. The first two algorithms rely on WWW meta-information, namely, the hyperlink structure, for ranking the WWW pages without considering term frequencies. The TFxIDF method is based on word occurrence statistics [12], whereas the Vector Spreading Activation method makes use of both word occurrence statistics and the hyperlink structure in ranking.

3.1 Terminology and Notations

In the WWW environment, a document is commonly referred to as a WWW page. In this paper, we use the term *page* and *document* interchangeably. The following notations are used in the rest of this paper.

M	:	the number of query words.
Q_j	:	the j-th query word, for $(1 < j < M)$.
N	:	the number of WWW pages in the index database.
P_i	:	the i-th WWW page or its ID number for $1 < i < N$.
$R_{i,q}$	:	the relevance score of P_i with respect to query q.
$Li_{i,k}$	:	the occurrence of an incoming hyperlink from P_k to P_i, where $Li_{i,k} = 1$ if such a hyperlink exists, or 0 otherwise.
$Lo_{i,k}$	:	the occurrence of an outgoing hyperlink from P_i to P_k, where $Lo_{i,k} = 1$ if such a hyperlink exists, or 0 otherwise.
$C_{i,j}$	:	occurrence of Q_j in P_i, where $C_{i,j} = 1$ if P_i contains Q_j, or 0 otherwise.

3.2 Description of the Algorithms

3.2.1 Boolean Spreading Activation

This algorithm is based on the Boolean retrieval model, where retrieval is based solely on the occurrence or absence of keywords in the documents. We extend the Boolean model so that documents can be ranked based on the number of query words they contain. This strategy can be considered as a simple fuzzy set retrieval model in contrast to the rigorous set membership of the Boolean retrieval model. More formally, document i is assigned a relevance score, $R_{i,q}$, with respect to query q as follows.

$$R_{i,q} = \sum_{j=1}^{M} C_{i,j} \tag{1}$$

Notice that term frequency is not used in the formula. It is assumed that the query does not contain any disjunctions nor negations. Disjunctions can be removed by normalization or splitting the query into separate conjunctive clauses. Negations can be removed by disqualifying all documents containing the negated terms prior to the ranking.

The Boolean Spreading Activation algorithm extends this strategy by propagating the occurrence of a query word in a document to its neighboring documents. This is possible in the WWW environment because a document can have hyperlink(s) to/from one or more other document(s), forming a network of documents. We assume that if two documents are linked to one another there must be some semantic relation(s) between the two. In other words, document P_i which does not contain query word Q_j but is linked to another document P_k containing Q_j is treated as if it contains Q_j. However, we assign P_i with a smaller score than if it actually contained Q_j. For each WWW page P_i, the algorithm assigns a relevance score with respect to query q as follows.

$$R_{i,q} = \sum_{j=1}^{M} I_{i,j} \tag{2}$$

where $I_{i,j}$ is defined as:

$$I_{i,j} = \begin{cases} c_1 & \text{if } C_{i,j} = 1 \\ c_2 & \text{if there exists } k \text{ such that} \\ & C_{k,j} = 1 \text{ and } Li_{i,k} + Lo_{i,k} > 0 \\ 0 & \text{otherwise} \end{cases}$$

c_1 and c_2 are constants $(c_1, c_2 > 0)$ where $c_1 > c_2$. The algorithm is not sensitive to the values of these two constants. We prove this point empirically in the next section. In the implementation, we use $c_1 = 10$ and $c_2 = 1$.

3.2.2 Most-cited

As with Boolean Spreading Activation, this algorithm takes advantage of information about hyperlinks between WWW pages. Each page is assigned a relevance score which is the sum of the number of query words contained in other pages citing, or having a hyperlink referring to, the page. More formally, the relevance score of page P_i with respect to query q is defined as:

$$R_{i,q} = \sum_{k=1,k\neq i}^{N} (Li_{i,k} \sum_{j=1}^{M} C_{k,j}) \quad (3)$$

The objective of this algorithm is to assign, among potentially relevant documents, larger scores to the referenced documents than to the referencing documents.

3.2.3 TFxIDF

The TFxIDF algorithm is based on the well known vector space model [12], which typically uses the cosine of the angle between the document and query vectors in a multi-dimensional space as the similarity measure. As described in [13], vector-length normalization can be applied when computing the relevance score, $R_{i,q}$, of page P_i with respect to query q:

$$R_{i,q} = \frac{\sum_{term_j \in q}(0.5 + 0.5\frac{TF_{i,j}}{TFmax_i})IDF_j}{\sqrt{\sum_{term_j \in P_i}(0.5 + 0.5\frac{TF_{i,j}}{TFmax_i})^2(IDF_j)^2}} \quad (4)$$

where:

$TF_{i,j}$	:	the term frequency of Q_j in P_i
$TF_{i,max}$	:	the maximum term frequency of a keyword in P_i
IDF_j	:	$\log(N/\sum_{i=1}^{N} C_{i,j})$

Generally speaking, the relevance score of a document is the sum of the weights of the query terms that appear in the document, normalized by the Euclidean vector length of the document. The weight of a term is a function of the word's occurrence frequency (also called the term frequency) in the document and the number of documents containing the word in the collection (i.e., the inverse document frequency). This weighting function gives higher weights to terms which occur frequently in a small set of the documents.

The full vector space model is very expensive to implement, because the normalization factor is very expensive to compute. In our TFxIDF algorithm, the normalization factor is not used. That is, the relevance score is computed by:

$$R_{i,q} = \sum_{term_j \in q} (0.5 + 0.5\frac{TF_{i,j}}{TF_{i,max}})IDF_j \quad (5)$$

In the next section, we show empirically that this simplified method works better in the WWW environment than the vector space model with vector-length normalization.

3.3 Vector Spreading Activation

This algorithm combines the vector space model and spreading activation model. In this algorithm, each document is first assigned a relevance score using TFxIDF algorithm, then the score of a document is propagated to the documents it references. More formally, the algorithm assigns a relevance score to page P_i with respect to query q as follows.

$$R_{i,q} = S_{i,q} + \sum_{j=1,j\neq i}^{N} \alpha Li_{i,j} \cdot S_{j,q} \quad (6)$$

where $S_{i,q}$ is the TFxIDF score of P_i as defined in equation 5. α $(0 < \alpha < 1)$ is a constant link weight. Through an experiment (discussed in the next section), we found that 0.2 is the optimal value of α.

4 Retrieval Effectiveness

We evaluated the four algorithms on an index database covering pages at the Chinese University of Hong Kong (`CUHK.HK` domain). CUHK site was chosen because of its reasonable size and it has a diverse collection of information provided by the university's various departments, from the fields of humanity to engineering. We froze the entire WWW collection by copying all of the WWW pages from the site to a local disk. This was done on April 26, 1995. We recorded 2393 WWW pages including 1139 non-HTML pages (non-inline image, sound, video, click-able map, CGI script, and other text files). We then built an index from the full-text collection as described before.

56 test queries[9] were used. The test queries were generated as follows. First, we selected at random 100 WWW pages from the collection. Of these 100 pages, we removed pages of directory type (indices, catalogs, hotlists, tables of contents, etc.) and non-HTML pages, resulting in 56 pages. Next, for each of these 56 pages, we manually extracted keywords from it, selected keywords which can be used to construct a phrase representing the central concept (topic) of the page, and constructed a query from that phrase. Boolean OR operators, along with scope markers, were used in the queries to specify synonyms.

Given a query, the judgment on whether a WWW page is relevant or not is somewhat ambiguous, as it is very subjective and may vary across users. For this evaluation, we define relevance in the context of resource discovery, i.e., a WWW page is considered relevant to a query if, by accessing the page, the user can find a resource address (URL) containing information pertinent to the query, or if the page itself is such a

[9] The test queries are listed in the Appendix of a report accessible at (http://dbx.cs.ust.hk:8000/doc/wwwindex.ps).

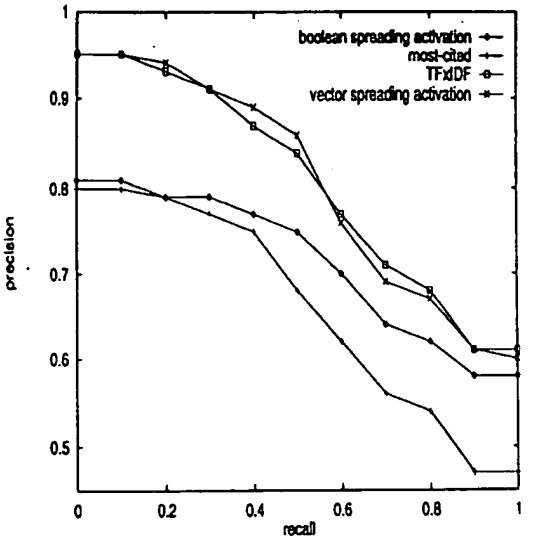


Figure 3: Recall-precision curve for each of the four search algorithms obtained by averaging the curves over the 56 test queries.

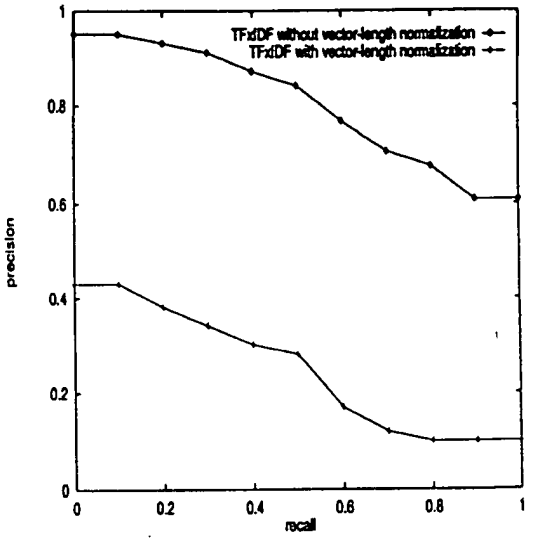


Figure 4: Recall-precision curve for TFxIDF with and without vector-length normalization obtained by averaging the curves over the 56 test queries.

resource. We manually examined the entire collection to identify the relevant WWW pages for each query.

We used the standard evaluation procedure [12] to compute the average interpolated recall/precision for each of the algorithms. The recall/precision curves are shown in figure 3. The high average precision obtained in this experiment is attributed to the query construction procedure which guarantees that each query has at least one highly relevant document. Figure 3 shows that Vector Spreading Activation has the best retrieval performance, followed by TFxIDF, Boolean Spreading Activation and Most-cited.

As mentioned in the previous section, we also conducted an experiment using the same 56 test queries to see whether vector-length normalization (see equation 4) could improve the retrieval effectiveness of TFxIDF. Figure 4 shows the recall-precision of TFxIDF with and without such a normalization. According to Salton and Buckley [13], vector-length normalization typically does not work well for short documents. This is consistent with our result since the average page length in the test collection is only 48.11 words (not including stop words and other words removed during the indexing process). It may be the case that vector-length normalization, in general, does not work for documents where the size of a segment with a coherent topic is small, e.g., a few sentences. In a WWW environment, it is common that a topic is only represented in a page by a hypertext anchor (click-able phrase).

To evaluate the sensitivity of Boolean Spreading Activation and Vector Spreading Activation algorithms to the choice of parameter values used, we conducted performance evaluation similar to the above on a range of parameter values.

Figure 5 shows the recall-precision curves of Boolean Spreading Activation on the 56 test queries using a number of c_1 and c_2 value combinations (see equation 2). By setting c_2 equals 0, that is by disabling the spreading activation effect, we obtained the retrieval effectiveness of the fuzzy set retrieval model (see equation 1). Enabling the spreading activation effect by setting $0 < c_2 < c_1$ improved the algorithm's recall. The algorithm produced the same recall-precision curve for (c_1, c_2) combinations of (2,1), (5,1), (10,1) and (10,5). Poor retrieval effectiveness resulted when c_1 was set equal to c_2, in which case many pages with various degrees of actual relevance had the same relevance score.

Figure 6 shows the recall-precision curves of Vector Spreading Activation on the 56 test queries with α parameter value of 0.0 (TFxIDF without the spreading activation effect), 0.1, 0.2, 0.3, 0.4 and 0.5 (see equation 6). The best retrieval performance was achieved with α equals 0.2.

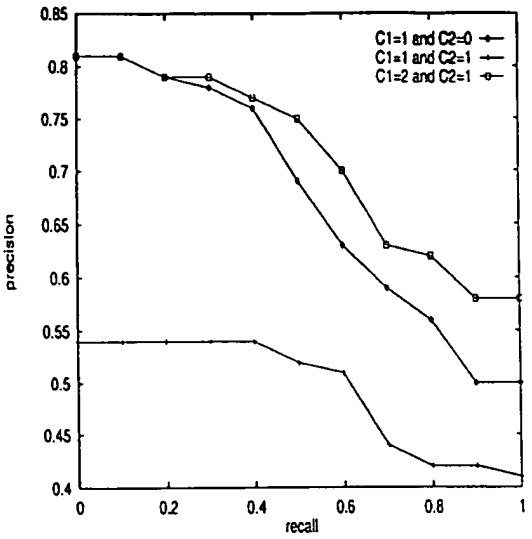


Figure 5: Recall-precision curves of Boolean Spreading Activation algorithm on the 56 test queries with (c_1, c_2) value pairs of (1,0), (2,1) and (1,1).

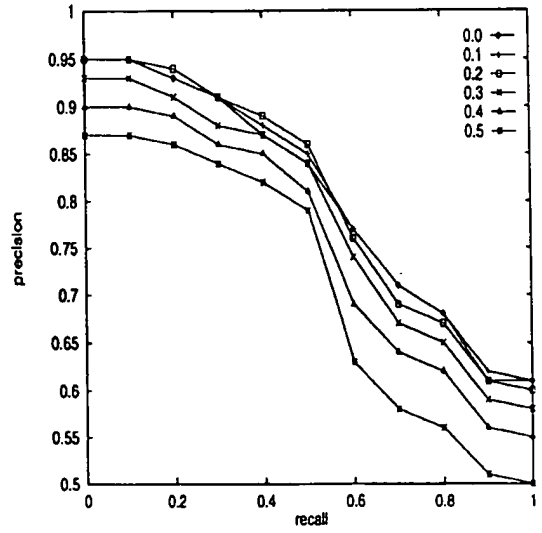


Figure 6: Recall-precision curves of Vector Spreading Activation algorithm on the 56 test queries with α value of 0.0, 0.1, 0.2, 0.3, 0.4 and 0.5.

5 Discussion and Future Work

5.1 Conclusion

The results of the experiment in Section 4 provide us with some hints on the nature of WWW information retrieval environment. The relatively superior retrieval effectiveness of TFxIDF and Vector Spreading Activation search algorithms shows that the concentration or distribution of words in a WWW page and across WWW pages is a good indicator of the page's contents or portions thereof. Algorithms which rely on meta-information such as hyperlinks information, while intuitive, did not perform as well. This shows that the interconnectivity between WWW pages is not a reliable indicator of semantic relationships between the contents of the linked pages. The poor overall performances of these ranking algorithms may also be attributed to the fact that many WWW pages contain many different and unrelated topics in a single page. This is true for many home pages, index pages, what's-new pages, hotlist, directory and catalog pages which occur frequently in the WWW environment. It is worth noting that, since most of the test queries are phrases taken out from actual pages with some synonyms added, all of the algorithms showed good recalls.

5.2 Other Index Servers

There are many robot-based WWW index and search services on the Internet today.[10] However, among the well known robots, only a few employ full-text indexing, e.g., WebCrawler[11] [11], the Repository Based Software Engineering Project Spider[12] (RBSE) [5], and Lycos.[13] Other services index only page titles and first level headings (e.g., JumpStation[14]), or titles, headings and anchor hypertexts (e.g., World Wide Web Worm or WWWW[15]). Our indexer robot takes other HTML tokens such as words in bold-face or italics, the first sentence of every list item, in addition to titles, all-level headings and anchor hypertexts. Our scheme is a balance between full-text and title-only schemes by taking advantage of HTML meta-information as much as possible. On a WWW page containing mostly lists such as an index page, our scheme extracts nearly as much words as a full-text scheme.

Not many index servers use sophisticated information retrieval models beyond a simple Boolean model or pattern matching based on Unix *egrep* (e.g., WWWW), with exception of the RBSE, the WebCrawler, and Lycos. The RBSE uses WAIS search

[10] A list of WWW robots can be found at (http://info.webcrawler.com/mak/projects/robots/active.html).

[11] (http://webcrawler.cs.washington.edu/WebCrawler/Home.html).

[12] (http://rbse.jsc.nasa.gov/eichmann/urlsearch.html).

[13] (http://lycos.cs.cmu.edu/).

[14] (http://www.stir.ac.uk/jsbin/js).

[15] (http://www.cs.colorado.edu/home/mcbryan/WWWW.html).

engine which ranks WWW pages based on the occurrence frequencies or term frequency (TF) of the query words [10]. The WebCrawler and Lycos, as with our index server, rank the pages based on term frequency and inverse document frequency (TFxIDF). As of this writing, we are not aware of any attempt to quantitatively measure the retrieval effectiveness of search algorithms in the WWW environment as in the present work.

5.3 Future Work

As part of an on-going research project, our next step in developing the WWW index server is to study the effectiveness of other information retrieval techniques, including relevance feedback and sophisticated user interface techniques. We are also working on developing methods based word distribution statistics, which has been proven useful in this paper, for automatic catalog generation, index database compression/summarization and multi-server indexing.

Acknowledgments

This research is supported by grants from the Sino Software Research Centre (SSRC), project no. SSRC94/95.EG01, and the Hong Kong Research Grant Council (RGC), project no. HKUST670/95E.

References

[1] Berners-Lee, T., "Uniform Resource Locators," *Internet Working Draft*, 1 January 1994.

[2] Berners-Lee, T., "Hypertext Transfer Protocol," *Internet Working Draft*, 5 November 1993.

[3] Berners-Lee, T., and Connolly, D., "Hypertext Markup Language," *Internet Working Draft*, 13 July 1993.

[4] Berners-Lee, T., Cailliau, R., Groff, J., and Pollermann, B., "World Wide Web: The Information Universe," *Electronic Networking: Research, Applications and Policy*, 1(2), 1992.

[5] Eichmann, D., "The RBSE Spider - Balancing Effective Search against Web Load," In *Proceedings of the First International Conference on the World Wide Web*, Geneva, Switzerland, May 1994.

[6] Gravano, L., Tomasic, A., and Garcia-Molina, H., "The Efficacy of GLOSS for the Text Database Discovery Problem," Technical Report STAN-CS-TR-93-2, Stanford University, October 1993.

[7] Koster, M., "ALIWEB: Archie-like Indexing in the Web," *Computer Networks and ISDN Systems*, 27(2), pp. 175-182, 1994.

[8] Krol, E., *The Whole Internet User's Guide & Catalog*, O'Reilly & Associates, Sebastopol CA, 1992.

[9] Nelson, P., "GDBM - The GNU Database Manager," online manual pages, Version 1.7.3, 1990.

[10] Pfeifer, U., Fuhr, N., and Huynh, T., "Searching Structured Documents with the Enhanced Retrieval Functionality of freeWais-sf and SFgate," *Computer Networks and ISDN Systems*, 27(7), pp. 1027-1036, 1995.

[11] Pinkerton, B., "Finding What People Want: Experiences with the WebCrawler," In *Proceedings of the First International Conference on the World Wide Web*, Geneva, Switzerland, May 1994.

[12] Salton, G., and McGill, M., *Introduction to Modern Information Retrieval*, McGraw-Hill, New York NY, 1983.

[13] Salton, G., and Buckley, C., "Term-Weighting Approaches in Automatic Text Retrieval," *Information Processing & Management*, 24(5), pp. 513-523, 1988.

[14] Schwartz, M., Emtage, A., Kahle, B., and Neumann, B., "A Comparison of Internet Resource Discovery Approaches," *Computer Systems*, 5(4), p461-493, 1992.

USING MULTIPLE EXAMPLES FOR CONTENT-BASED IMAGE RETRIEVAL

J. Assfalg, A. Del Bimbo, P. Pala

University of Florence
Dipartimento di Sistemi e Informatica
Via S.Marta 3, 50139 Firenze, Italy
{assfalg,delbimbo,pala}@dsi.unifi.it

ABSTRACT

Query specification for content-based image retrieval is typically accomplished through query-by-example paradigms, such as query-by-image and query-by-sketch. In some cases query-by-sketch can be difficult —lack of sketching abilities, difficulty to detect distinguishing image features— and querying is therefore performed through the query-by-image paradigm. However, this paradigm often fails since a single sample image rarely includes *all* and *only* the characterizing elements the user is looking for.

This paper prsents a system that supports query-by-image using multiple image examples, both *positive* and *negative*. The system also enables editing of examples so as to disregard those image features that are not relevant to the query.

1. INTRODUCTION

Content Based Image Retrieval (CBIR) is an extension to traditional information retrieval that supports querying of images through a visual specification, thus addressing perceptual content of visual information [1]. Typically, queries are performed through the query-by-example paradigm which requires the user to provide a representation of the *visual concept* characterizing the searched image.

In modern CBIR systems, querying-by-example mainly takes place in two different forms: query-by-image and query-by-sketch. Query-by-image allows the user to take a sample image —either from a sample image set or from the answer to a previous query— and use it as a prototype to retrieve images with similar content. This paradigm supports retrieval by global image similarity addressing global color/texture distribution, and the overall image structure. Query-by-sketch allows the user to sketch contours of salient regions on a white-board. Regions can be manually authored or traced from an image, following the objects' contours. Once they have been sketched, regions can be characterized by color, texture, shape, position, and area. This paradigm supports retrieval based on local properties of images.

Although it is widely employed in current systems for content-based retrieval, query-by-sketch has several drawbacks. The main drawback being that users find it difficult to create by scratch suitable images embodying the visual concept to be searched for. This is usually due to the lack of visual memory of most users, lack of sketching abilities, and difficulty of detecting those distinguishing salient features that actually characterize that concept. Due to this, examples are built —in most cases— according to the query-by-image paradigm. Existing images, either retrieved with a previous query or selected from a random subset of the database, are used as examples. A limiting factor, in this case, is that a single sample image rarely includes all and only the characterizing elements of a visual concept. On the one hand, a sole example typically includes a subset of the characterizing features. On the other hand, some features included in the example may not contribute to the definition of the visual concept. The former issue can be addressed by querying through multiple sample images. The latter requires that content editing of each example be supported by the system.

Furthermore, specification of a visual concept can be achieved by submitting several examples, providing positive as well as negative instances of that concept. Usage of positive and negative examples has been exploited for interactive learning of image classification ([6], [7], [9]). Examples are used to associate low-level features with high-level concepts to be used at query time.

Positive and negative examples have also been used to support relevance-feedback interaction ([4], [5], [10]) which allows the user to score the relevance of retrieved images. Relevance scores are used to change the system similarity measure so as to converge to a result that matches user's expectations. This prevents the use of

0-7803-6536-4/00/$10.00 (c) 2000 IEEE

any indexing structure that develops on a predefined similarity measure.

In this paper a system is presented that supports efficient querying by *Positive* and *Negative* examples. The system features a color-based retrieval engine. Histograms encode color content and an M-tree structure is used for indexing. Querying is performed through a visual interface which lets the user specify positive and negative examples.

The paper is organized as follows: Section 2 provides insight on content representation through histograms, and related properties; then, Section 3 expounds our solution for retrieval by positive and negative examples; Finally, in section 4 experimental results are reported and conclusions are drawn.

2. CONTENT REPRESENTATION THROUGH COLOR HISTOGRAMS

A generic histogram H with n bins is an element of the histogram space $\mathcal{H}^n \subset \mathbb{R}^n$.

Given an image and a discretization of a feature space, histogram bins count the number of occurences of points of that feature space in the image. Histograms provide a synthetic representation for content, and have been used for different features, such as color and shape ([2] [11]).

Histograms also support a multi-resolution description of image features. Given a partitioning of an image into n fine-grained regions, histograms provide a representation for the content of each of these regions. The representation of wider regions, at a less fine-grained level can be computed by merging the histograms of each region $\mathcal{R}_k^i$ at level i that contributes to the region $\mathcal{R}^{i+1}$ at level $i+1$ ($\mathcal{R}^{i+1} = \cup_{k=1}^{m} \mathcal{R}_k^i$). Each j-th bin h_j^{i+1} at level $i+1$ is computed as follows:

$$h_j^{i+1} = \sum_{k=1}^{m} h_{j,k}^i \qquad \forall j = 1, \ldots, n \tag{1}$$

It is also possible to provide lower resolution descriptions based on a coarser discretization of the feature space, with dimension $\tilde{n} < n$.

In order to compute the similarity between two histograms, a norm must be defined in the histogram space. This is accomplished through the introduction of a positive definite, symmetric *distance matrix* $A \in \mathbb{R}^{n \times n}$ that allows the distance between two histograms H and H' to be computed as:

$$\begin{aligned} \mathcal{D}(H, H') &= (H - H')^T A (H - H') = \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n} (h_i - h'_i)(h_j - h'_j) a_{ij} \end{aligned} \tag{2}$$

being h_i (h'_i) the i-th element of H (H').

This expression evidences that elements a_{ij} weight the extent to which the difference between the i-th and j-th bins contributes to $\mathcal{D}$. If the distance matrix A is a diagonal one, only differences between corresponding bins contribute to the computation of $\mathcal{D}$. However, if A isn't a diagonal matrix, elements a_{ij} ($i \neq j$) can be used to model the cross-distance between non-corresponding elements. This is particularly useful in order to partly recover from the loss of information associated with the discrete nature of content representation through histograms.

For the implementation of the system presented in this paper, 39 reference colors were selected to discretize the color space, and histograms with 39 bins are used to encode color information for each of the tiles images are partitioned into. Histograms are normalized with respect to the image size so as to provide scale invariance of the representation.

A distance matrix A_C has been defined for the computation of the distance between color histograms and structure histograms, respectively. The color distance matrix A_C is a 39×39 symmetric matrix whose elements a_{ij} encode the perceptual similarity between i-th and j-th reference color.

3. COMBINING MULTIPLE EXAMPLES INTO A SINGLE QUERY

The basic idea underlying our approach for querying by positive and negative examples is illustrated in Figure 1.

Let $\{P^i\}_{i=1}^n$ be the set of positive sample histograms and $\{N^i\}_{i=1}^m$ be the set of negative sample histograms. We want to represent the query requirements expressed through the positive and negative samples using a single composite query histogram H. The composite query histogram H must be close to all the elements of $\{P^i\}_{i=1}^n$ and far from all the elements of $\{N^i\}_{i=1}^m$. Hence H can be found as the histogram which minimizes the following functional:

$$\mathcal{F} = \sum_{i=1}^{n} \mathcal{D}(H, P^i) - \sum_{j=1}^{m} \mathcal{D}(H, N^j) \tag{3}$$

being $\mathcal{D}(\cdot,\cdot)$ the histogram distance function. That is:

$$\mathcal{D}(H, P^i) = \sum_r \sum_s (H_r - P_r^i)(H_s - P_s^i) a_{rs} \tag{4}$$

$$\mathcal{D}(H, N^j) = \sum_r \sum_s (H_r - N_r^j)(H_s - N_s^j) a_{rs} \tag{5}$$

Since the solution to Eq.(3) must be a regular histogram, Eq.(3) must be minimized subject to two distinct constraints. These are related to the norm of the

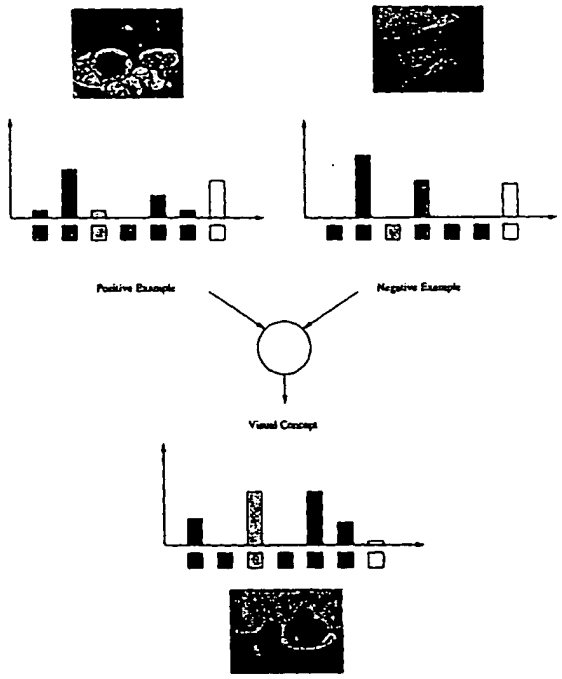


Figure 1: A number of examples, both positive and negative, is used to describe the visual concept. The system combines histograms of the examples to derive a single composite query histogram representing the visual concept. An image embodying the visual concept is also shown.

solution (the histogram must be a normalized vector) and to the positiveness of individual histogram components.

Thus, the composite query histogram $H = (H_1, \ldots, H_r)$ is the solution of an optimization problem with bilateral and unilateral constraints:

$$\begin{cases} \min_{H} \mathcal{F} \\ \sum_{i=1}^{r} H_i = 1 \\ H_i \geq 0 \qquad \forall i = 1, \ldots, r \end{cases} \tag{6}$$

The resulting histogram summarizes the positive and negative examples, and could be thought of as the histogram of a query image featuring the visual concept that is the object of the user's query.

4. A PROTOTYPE IMPLEMENTATION

The user interface is shown in Figure 2. The interface is composed of two main parts. The lower part includes the output image viewer. This displays images that are either randomly selected (obtained through the *Randomize* button) or retrieved as the result of a query session. The upper part of the interface includes two panels on the left and on the right —used to collect positive and negative examples, respectively—, and one editing area in the middle.

Images can be selected from the output viewer and added either to the positive or negative sample sets (through the *Add Positive* and *Add Negative* buttons, respectively).

A simple query with one positive sample image is shown in Figure 2. Retrieved images are displayed in the lower part of the interface. Similarly to the query image, all retrieved images feature bright and saturated red, cyan and green colors.

In Figure 3 a query refinement is shown. One of the retrieved images (3^{rd} row, 4^{th} column in Figure 2) is added to the negative sample set. The regions of this image that feature white and blue patterns are labeled as not relevant, and only regions with a green pattern are retained. The content of the positive sample image is edited as well. Image regions featuring a green pattern are labeled as not relevant. Retrieved images are shown in the lower part of the interface: All the images feature bright red and/or cyan patterns. No image features relevant green patterns. It can be noticed that the image used as positive example is not retrieved in the first position (it is ranked in the 3^{rd} position). This is indeed consistent with the query that includes a negative example with a large green region. As a result, the ranking score of the image used as positive example —that features a green region on the right— is penalized.

5. REFERENCES

[1] A. Del Bimbo, "Visual Information Retrieval", Academic Press, London 1999.

[2] M. Flickner, et al., "Query by Image and Video Content: the QBIC System", *IEEE Computer*, Vol. 28, n. 9, pp. 310-315, Sept. 1995.

[3] J.R. Bach, et al., "The Virage Image Search Engine: an Open Framework for Image Management", *Proc. SPIE Storage and Retrieval for Still Image and Video Databases*, vol. 2760, 1996, pp. 76-87.

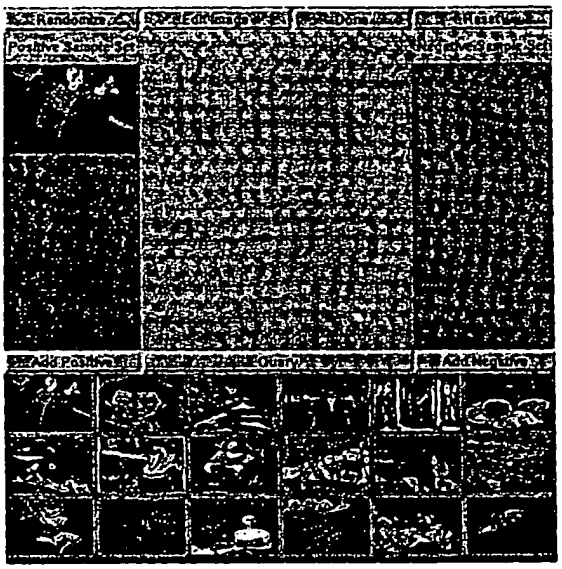


Figure 2: The user interface provides two panels for collecting positive and negative examples. An additional panel supports detailed content editing to specify which regions are significant to specify the visual concept to be searched for. Results of the query are shown in the lower panel of the interface.

Figure 3: The query of the example in Figure 2 is refined. An image of the previous result set is selected and, after some content editing, a negative example is added to the query specification. Images that match the new visual concept are shown at the bottom.

[4] L. Taycher, M. La Cascia and S. Sclaroff, "Image Digestion and Relevance Feedback in the ImageRover WWW Search Engine", *Proc. Visual '97*, San Diego, CA, Dec. 1997.

[5] Y. Rui, T.S. Huang, M. Ortega, S. Mehrotra, "Relevance Feedback: A Power Tool for Interactive Content-Based Image Retrieval", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 8, n. 5, September 1998.

[6] P. Lipson, E. Grimson, P. Sinha, "Configuration Based Scene Classification and Image Indexing", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'97)*, 1997.

[7] T.P. Minka, R.W. Picard, "Interactive Learning Using a 'Society of Models' ", *Pattern Recognition*, 30(4), 1997.

[8] A. Nagasaka, Y. Tanaka: "Automatic video indexing and full video search for object appearances", *IFIP Trans., Visual Database Systems II*, pp. 113-127, Knuth, Wegner (Eds.), Elsevier, 1992.

[9] A.L. Ratan, O. Maron, W.E.L. Grimson, T. Lozano-Pérez, "A Frameork for Learning Query Concepts in Image Classification", *Proccedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'99)*, Fort Collins (CO), June 1999.

[10] S. Santini, R. Jain, "Beyond Query by Example", *Proceedings of the Sixth ACM International Multimedia Conference, ACM Multimedia '98*, Bristol, England, September 1998.

[11] A. K. Jain and A. Vailaya. Image retrieval using color and shape. *Pattern Recognition*, 29(8):1233–1244, Aug. 1996.

QUERY BY IMAGE CONTENT USING MULTIPLE OBJECTS AND MULTIPLE FEATURES: USER INTERFACE ISSUES

Denis Lee *Ron Barber* *Wayne Niblack*
Myron Flickner *Jim Hafner* *Dragutin Petkovic*

IBM Almaden Research Center, San Jose, CA 95120

ABSTRACT

On-line collections of images are growing larger and more common, and tools are needed to efficiently manage, organize, and navigate through them. We have developed a prototype system called QBIC which allows complex multi-object and multi-feature queries of large image databases. The queries are based on image content – the colors, textures, shapes, and positions of images and the objects/regions they contain. The system computes numeric features to represent the image properties and uses similarity measures based on these features for image retrieval. The focus of this paper is its user interface which allows a user to graphically pose and refine queries based on multiple visual properties of images and their objects.

1. INTRODUCTION

Today's hardware technology allows us to acquire, store, manipulate, and transmit large numbers of digital images, and to generate large on-line image collections. Current systems for image management, retrieval, and database functions rely almost exclusively on keywords or text associated with each image for their access method. To complement the text-based methods, we are studying content-based approaches that: (1) are based on computed features such as color, texture, shape, and position of images and the objects they contain; (2) use similarity measures and retrieve images "like" or similar to a given image (returning them in ranked order); (3) allow complex queries in the sense that query predicates can specify multiple features of multiple objects in a scene ("a red round object on the left and large blue textured region on the right"); and (4) handle queries that are posed visually and graphically, such as by painting an approximate query image or picking a sample texture from a palette of textures. The emphasis of this paper is the graphical user interface we have developed that allows users to pose these queries and the facilities it provides to construct "multi-queries" involving global images features as well as multiple features of multiple objects.

Querying image databases is an active area of research. A survey of systems with query-by-content capabilities is given in [1]. QBIC has been presented in [2, 3, 4, 5, 6], and other recent systems are described in [7, 8]. Previous graphical user interface methods include placing icons that represent semantic objects such as house and car in a query area [9] and its extension to 3D using a manipulating glove to place icons in space to represent an image to be retrieved [10].

2. QBIC QUERIES

The QBIC system distinguishes between "scenes" (or images) and "objects". A scene is a full color image or single frame of video and an object is a part of a scene, e.g., a person in a beach scene. Each scene has zero or more objects. Objects are identified manually or semi-automatically. QBIC computes the following features:

Objects	Images
1. average color	1. average color
2. histogram color	2. histogram color
3. texture	3. texture
4. shape	4. positional edges (sketch)
5. location	5. positional color (draw)

Further descriptions of these features are given in [2]. The features, precomputed and stored in a database for subsequent queries, are vectors of numeric values. For example, the average color feature is a 3 element vector and the shape feature is a 20 element vector.

Once the features for objects and images have been computed, queries may then be executed. Queries may be object-based, in which a user requests images containing objects with certain features, scene-based, in which full scenes with certain features are retrieved, or a combination of both. For example, an object-based query might search for images containing square red

0-8186-6950-0/94 $4.00 © 1994 IEEE

objects. A scene-based query might search for images (scenes) having a certain percentage of the colors red and blue. Query results are based on the similarity of database items to the query items (either objects or scenes) and use similarity functions for each feature. These similarity functions are normalized so that they can be meaningfully combined. Most of the similarity functions are based on the weighted Euclidean distance in the corresponding feature space (e.g., three dimensional average color), although some, for example, color histograms, have their own similary function. See [2] for details.

3. USER INTERFACE

A query specification can include color, texture, shape, location, and/or text features of a single object or scene or, in the case of "multi-queries", of scene features together with multiple objects, each of which have multiple features.

The user interface to support such queries has two main parts: the Query Specification windows to pose the queries and the Query Results window to display the results.

3.1. The Query Specification Windows

The Query Specification Windows form a hierarchy of three levels. The top level is the symbolic representation of the query, the middle controls the enabling and weighting of feature values, and the lowest allows the feature values to be set or selected.

Level 1: The Query Window. An example of the top level, or Query Window, is shown in Figure 1. The window has a menu bar and a query area. The user can specify an object or a scene component of a query by selecting appropriately from the Create option on the menu bar, which in turn, places an "object" or "scene" icon into the query area. An icon with a rectangle represents a scene and one without a rectangle represents an object. Multi-queries are formed by creating multiple icons in the query area. By clicking on the icon, the user can get a menu which contains the Copy (replicate the icon and all of its features), Delete, Move, or Edit options. This last option causes the appearance of the corresponding Object or Scene Feature Windows in Figures 2 and 3, which form the middle level of the query hierarchy.

Level 2: The Feature Windows. These windows contain all the features available for object or scene queries. Each available feature has an enable button, a feature type field, editor/sampler radio buttons, a value/picker button, and a weight slider. The enable button signifies that the feature is to be used in

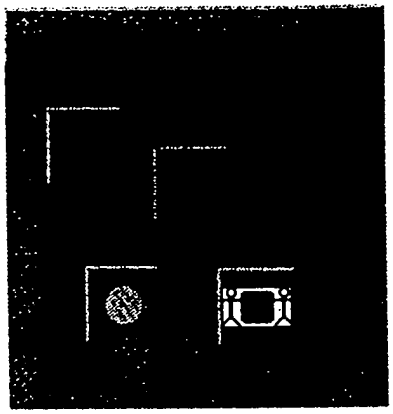


Figure 1: The query window.

queries. The weight of the feature is determined by a slider which goes from 0 to 1 allowing the user to adjust the relative importance of the feature during queries. Each feature can be set either by using an editor or by selecting a pre-defined value from a "sampler". The editor/sampler radio buttons determine whether to bring up an editor or sampler whenever the associated value/picker button is pushed. When a feature value of an object or scene is set, the value/picker button is modified to represent that value. For example, when a color is selected, it is painted on the color value button; when a shape is drawn, it is reproduced on the button, etc. Notice in Figure 2 that the buttons match the properties selected in Figures 4, 5, 6, and 7.

Level 3: The Editors and Samplers. To set the feature values, the user clicks on the value buttons in the Feature Windows. Depending on which of the Editor or Sampler radio buttons is enabled, the user gets either an Editor or a Sampler. Editors allow any values to be set/drawn and Samplers allow one value from a given set to be chosen. The editor for color, Figure 4, uses a standard set of (R, G, B) sliders and a color patch area which shows the current selected color. The histogram color editor, Figure 5, allows multiple colors and their amounts to be selected. The palette on the left of the picker displays the set of selectable colors. Once a color is selected, it will appear in a box on the right and the relative amount of the color can be specified by the sliders.

Shape, sketch, and draw use the same editor, Figure 6, which incorporates a palette of colors on the left and a drawing area on the right. In between, the currently selected color is shown, together with a row of buttons: Polygon, Ellipse, Rectangle, and Line, for the drawing mode desired. The user clicks on a color in the palette

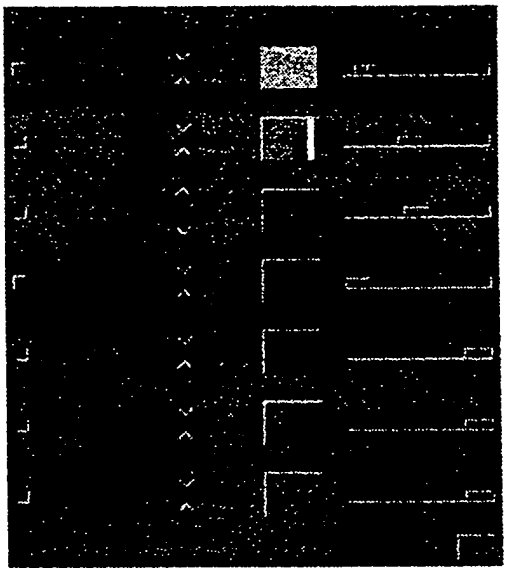


Figure 2: The object feature window lets the user enable, disable, and weight object features and call up the associated pickers.

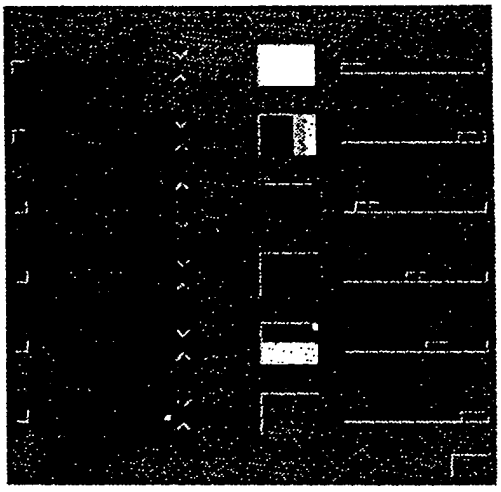


Figure 3: The scene feature window lets the user enable, disable, and weight scene features and call up the associated pickers.

to choose a color, selects a drawing mode, and draws in the drawing area. An erase mode is also available.

The location editor, Figure 7, is one way the user can specify an (x, y) location of an object in a query. (The other is by positioning the icon within the query window.) The picker is a simple blank area with an "X" marking the spot where the object should be located. Lastly, the text editor is a simple window for entering text.

A Sampler displays a set of samples from which one can be chosen. An example is the texture sampler shown in Figure 8. Multiple samplers for a given feature are possible and are arrange hierarchically, thereby, allowing, for example, texture samplers arranged as ALL-TEXTURES containing the two subsets NATURAL and SYNTHETIC.

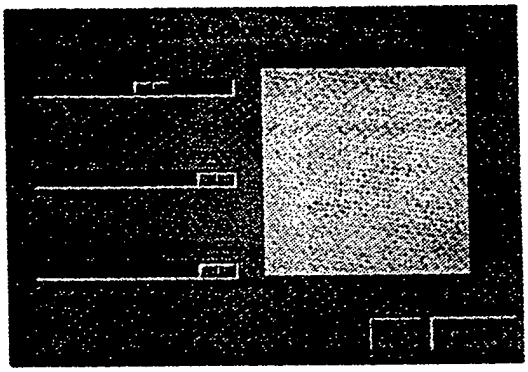


Figure 4: The RGB color picker.

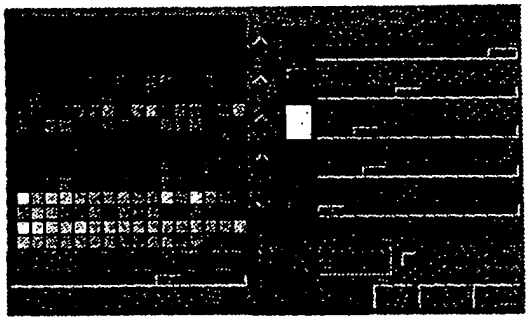


Figure 5: The multicolor color picker.

Once the features of a scene or object are set, its icon in the query area will display symbols representing each of the features enabled for querying as shown in Figure 1. The color feature is symbolized by filling the icon with the selected color. A small RGB histogram

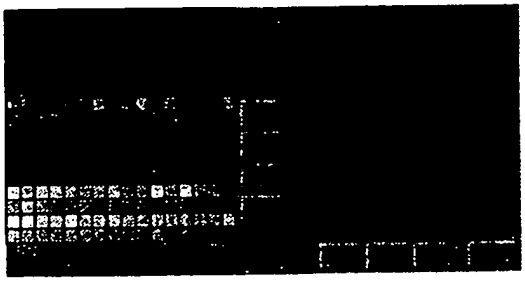


Figure 6: The freehand drawing window for specifying an object shape or scene sketch.

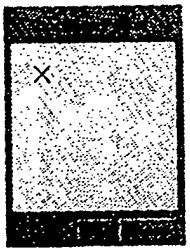


Figure 7: The location picker.

Figure 8: A texture Sampler.

on an icon represents the histogram color feature. The symbol for the texture feature is a hash pattern. The text feature is represented by the letter "T". The location feature in an object is symbolized by four arrows at each corner of the icon. The direction of the arrows indicates whether the location of the icon in the query area (arrows pointing inward) or the location in the location picker (arrows pointing outward) is used. The shape feature of an object is shown on an object icon as a circle. In a scene icon, the sketch feature is represented by two stick figures at each end of the rectangle.

Mutli-queries can easily be posed using this interface. For example, one can imagine a query with a red circular object, a blue object in the center of an image, and a green area at the lower portion of the scene. The query area would have three icons with the appropriate symbols representing the features of the two objects and the scene. By editing the objects or scene, their features for the query are shown in greater detail such as the actual circular shape or the actual portion which is green.

3.2. The Query Results Window

After a query has been specified, it is run by selecting Query on the menu bar of the Query Window and then Execute. The similarity measures, one for each feature of each item in the query, are combined into a final similarity measure. The combined results are obtained by normalizing all individual similarity measures by their variance and combining them in a weighted sum.

The ordered results of a query are displayed in the Query Results Window from best match to nth best match (n is user-settable and we usually display the top 20). Each image returned is displayed as a reduced "thumbnail". This thumbnail is active and can be clicked on to give a menu of options. The options include queries of the form "Find images like this one", display the similarity value of this image to the query image, display the (larger) full scale image, place the image in a holding area for later processing, perform a user defined image operation or comparison, and so on.

The Query Results Window is divided into two areas, the Results Area and the Holding Area. Each time a query is executed, the retrieved image thumbnails are displayed in the Results Area, overwriting any previous results. The user can move any thumbnail or set of thumbnails to the Holding Area. This process saves them so that they can be compared with later query results, used as inputs to subsequent queries, etc. Any menu operations available on thumbnails in the Results Area (display, find images like this one, etc.) can also be applied to thumbnails in the Holding Area.

4. RESULTS

Results from a sample query are shown in Figure 9. The same results, clipped to the bounding box of the area containing the queried objects, are displayed in Figure 10. The query was a multi-query for images with two objects. The first object was specified using the color picker to select the color red and the shape picker to draw a round shape. Similarly, the second object was defined to have a green color and round shape.

Figure 9: Results from a query.

Figure 10: Results from the same query as above. In this view, the results are "clipped" to the bounding box of the image containing the objects that participated in the query.

5. CONCLUSION AND FUTURE WORK

Databases containing large on-line image collections are becoming common and tools are needed to manage, organize, and retrieve images from these databases. A user interface allowing complex queries and query refinement is an important component of a query/browse tool. The QBIC system provides such an interface and lets the user pose complex visual queries based on both global scene properties and properties of multiple objects within the scenes. Qbic was implemented in C and X11/Motif on an IBM Risc/6000.

Areas of future work include combining QBIC with more powerful text search methods, adding logical operations such as AND's, OR's, and NOT's to the specification of a query, developing new features and their corresponding similarity matching algorithms, and presenting all this functionality through a well-designed and powerful user interface.

6. REFERENCES

[1] A. E. Cawkell. Picture queries and picture databases. *Journal of Information Sciences*, 19(6):409–423, 1993.

[2] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, and P. Yanker. The QBIC project: Querying images by content using color, texture, and shape. In *SPIE Conference 1908, Storage and Retrieval for Image and Video Databases*, Feb. 1993.

[3] R. Barber, W. Equitz, C. Faloutsos, M. Flickner, W. Niblack, D. Petkovic, and P. Yanker. Query by content for large on-line image collections. In *Multimedia Systems, An IEEE Tutorial*. IEEE Press, 1994.

[4] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3:1–28, 1994.

[5] B. Scassellati, S. Alexopoulos, and M. Flickner. Retrieving images by 2d shape: A comparison of computation methods with human perceptual judgments. In *SPIE Conference on Storage and Retrieval for Image and Video Databases*, volume 2185, Feb. 1994.

[6] D. Lee, R. Barber, W. Niblack, M. Flickner, J. Hafner, and D. Petkovic. Complex queries for a query-by-content image database. In *ICPR*, Jerusalem, Oct. 1994. IAPR.

[7] H. Sakamoto, H. Suzuki, and A. Uemori. Flexible montage retrieval for image data. *Proceedings: Storage and Retrieval for Image and Video Databases II*, pages 25–33, San Jose, CA, 7-8 Feb. 1994. SPIE.

[8] Y. Gong, H. Zhang, H. C. Chuan, and M. Sakauchi. An image database system with content capturing and fast indexing abilities. In *International Conference on Multimedia Computing and Systems*. IEEE, May 14-20 1994.

[9] S. K. Chang and Erland Jungert. Pictorial data management based upon the theory of symbolic projections. *Journal of Visual Languages and Computing*, 2:195–215, 1991.

[10] A. Del Bimbo, M. Campanai, and P. Nesi. 3-d visual query language for image databases. *Journal of Visual Languages and Computing*, 3:257–271, 1992.

HyperSQL: Web-based Query Interfaces for Biological Databases*

Mark Newsome[1], Cherri Pancake[1], and Joe Hanus[2]
Department of Computer Science[1], Department of Botany and Plant Pathology[2]
{newsome | pancake}@research.cs.orst.edu, hanusj@mgd.cordley.orst.edu
Oregon State University, Corvallis, OR 97331
http://mgd.cordley.orst.edu/hyperSQL

*This work is sponsored by NSF Grants BIR 9503712 and 9402192.

Abstract

HyperSQL is an interoperability layer that enables database administrators to rapidly construct browser-based query interfaces to remote Sybase databases. Current browsers (i.e., Netscape, Mosaic, Internet Explorer) do not easily interoperate with databases without extensive "CGI" (Common Gateway Interface) programming. HyperSQL can be used to create forms- and hypertext-based database interfaces for non computer experts (e.g., scientists, business users). Such interfaces permit the user to query databases by filling out query forms selected from menus. No knowledge of SQL is required because the interface automatically composes SQL from user input. Database results are automatically formatted as graphics and hypertext, including clickable links which can issue additional queries for browsing through related data, bring up other Web pages, or access remote search engines.

Query interfaces are constructed by inserting a small set of HyperSQL descriptors and HTML formatting into text files. No compilation is necessary because commands are interpreted and carried out by our special gateway, positioned between the remote databases and the Web browser. Feedback from developers who have used the initial release of HyperSQL has been encouraging. At present, query interfaces have been successfully implemented for three major NSF-sponsored biological databases: Microbial Germplasm Database, Mycological Types Collection, and Vascular Plants Types Collection.

1. Motivation

Accessing information stored in biological databases helps speed research and is essential to solve complex problems requiring data from multiple sources. Such databases are not likely to be centrally located, because they are typically maintained under the control of individual scientists/collectors who know the limitations of the data.

Many large-scale biological collections are stored in Sybase databases (e.g., Genome Database, Microbial Germplasm Database, National Fungus Collection, Nature Conservancy Database). Sybase is a commercial relational database management system that employs SQL (Structured Query Language). SQL was adopted as the standard query language in the scientific community because it is an industry standard, is well-defined and has a solid theoretical foundation in set theory, and is implemented in successful and proven commercial products (i.e., Sybase, Oracle, Informix) that run on high-end UNIX workstations capable of managing large volumes of data.

Researchers in the biological sciences have not had an easy time retrieving information stored in SQL databases. There are four primary reasons for this difficulty. First, SQL requires that users remember too much information about database organization. Before users can formulate queries, they must remember or look up the names of tables and data items, which requires knowledge of the database's proprietary command language. Moreover, users must determine the logical meanings of the data items, which can be frustrating because of cryptic naming conventions and the use of synonyms. Often there is little more available than an uncommented listing of the database schema. These problems are exacerbated when tables in the database contain a large number of data categories.

Second, users are forced to use cryptic command-line tools (i.e., Sybase isql) intended for database administrators. These are based on proprietary command languages[1], present information in a low-level format, and are unforgiving of mistakes. Furthermore, to access

[1] Although the vendors of the most commonly used SQL databases (i.e., Sybase, Oracle, Informix) adhere to the SQL standard, they use incompatible tools, programmatic interfaces, and command languages.

1060-3425/97 $10.00 © 1997 IEEE

databases at remote sites, scientists must learn additional connection utilities (e.g., telnet, ftp).

Third, SQL itself is terse, error prone, and unfriendly. It is easy to misspell a keyword or data name, omit a "join" clause, or worse, construct a query that returns unintended results [Sme95]. We believe it is senseless to expect scientists—who have neither the time nor the desire to be experts in computer science—to become proficient in SQL programming and database theory.

Last, there is a lack of meaningful feedback during the query process [JV85]. The chances of a casual user formulating a query correctly on the first attempt are slim. With command-line SQL, the user must bear the burden of determining the exact nature of an error. Messages are cryptic and generally do not reveal the source of the problem.

Since SQL's inception in the 1970s, database vendors have virtually ignored the scientific community, instead building support tools for business and financial users (i.e., inventory, accounting, stock analysis). The need for query assistance tools targeting scientific users has been expressed by researchers at number of biocomputing workshops and conferences [FJP90, ZI94]. To fill the need, we worked closely with scientists from the Department of Botany and Plant Pathology to develop HyperSQL.

HyperSQL is a development tool for constructing Web-based query interfaces to SQL databases. Our goal is to make it easier for scientists to retrieve information from remote scientific databases using common Web browsers (Netscape, Mosaic, Internet Explorer), which lack the capability to interoperate directly with SQL databases. Our software makes it possible to layer browser-based query interfaces on top of normal SQL. No modifications to either the Web browser or the SQL database are required.

With our software, forms and hypertext-based interfaces eliminate direct exposure to SQL and low-level database tools. The user queries SQL databases by entering search criteria into forms. No knowledge of SQL is required because the interface automatically generates SQL queries from the user's input. Since the form is largely independent of SQL, it can use discipline-specific (rather than computer-related) terminology. The results of the query are formatted as hypertext, so they can include hyperlinks that can be selected to browse the database for related information, bring up other Web pages, or access remote search engines.

For query assistance tools, browsers offer several important features. They employ a conceptually simple hypertext model that even users with little or no computer training can easily learn and use. The browser serves to replace a number of low-level tools (i.e., telnet, ftp, gopher) and provide unified means of displaying data in a variety of formats (i.e., text, tables, images, sound, movies); furnish the features necessary to create database front-ends—tables and forms (i.e., text boxes, push buttons, radio lights, pulldown menus, scrolled lists)—and operate independently of both location and computer platform. Finally, they are available for free or at a low cost. The remainder of this paper is organized as follows. Section 2 demonstrates how HyperSQL interfaces are used; the following section shows how they are constructed. Section 4 shows the organization of query files, while Section 5 discusses the architecture and how HyperSQL is implemented. Section 6 summarizes related work by others. The last section discusses the current status of HyperSQL and our future plans.

2. How HyperSQL Query Interfaces are Used

HyperSQL query interfaces are constructed by inserting a small set of HyperSQL descriptors [NPHM96] and HTML (Hypertext Markup Language) [BLCL+94] into text files. No compilation is necessary because commands are interpreted and carried out by our special gateway, positioned between the remote databases and the browser software. Operation of the gateway is transparent and users need not be aware of its presence.

A HyperSQL query interface is composed of a **query menu**, **query form**, **results screen**, and one or more **browse screens** (Figure 1). The **query menu** serves as a starting point for users, presenting introductory information about the database and displaying a list of queries, organized by subject. Clicking on one of the query selections brings up a **query form**. HyperSQL automatically prompts the user for a password if one is required to gain access to the database.

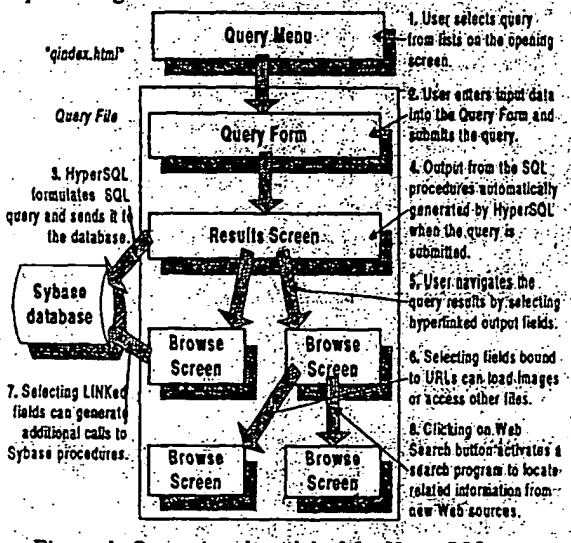


Figure 1. Operational model of the HyperSQL query interface

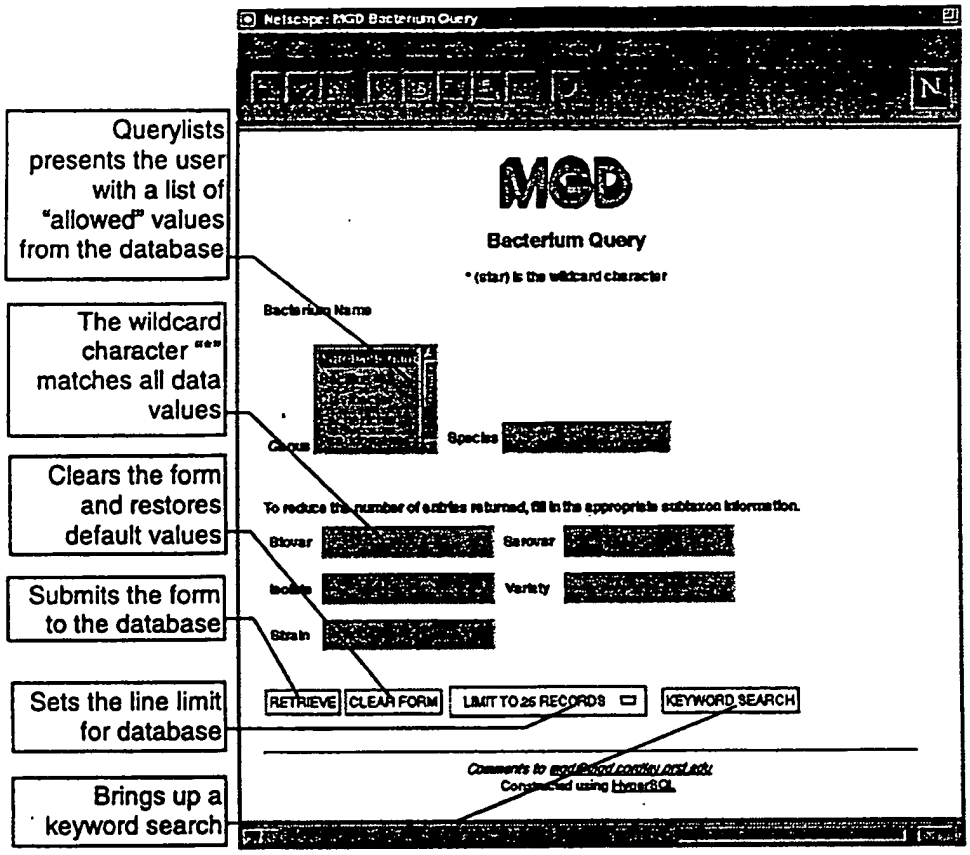


Figure 2. Example of a query form (shown in a Netscape browser)

The user enters search criteria on the query form (Figure 2). He/she can fill in as much information as available, or none at all; by default all text fields are supplied with wild-cards ("*"), meaning "match everything." A row of standard buttons are automatically provided at the bottom of the form. The `Retrieve` button submits the form and returns results. `Clear` erases any information entered into the form, replacing them with default values.

The pulldown menu located next to the `Clear` button is used to limit the number of results returned from the query; by default output is limited to 25 records. Clicking the `Keyword Search` button brings up a form that accepts a text string to be located by searching all data fields. For later reuse, forms containing favorite or frequently used sets of values can be saved by selecting "`Add to Bookmarks`" (or equivalent) in the browser.

HyperSQL query forms supports "query refinement," allowing the user to fill in fields incrementally by selecting from "querylists" (lists of allowed values). Querylists are useful when the user does not know what the database contains, or simply wants to select from a list of values rather than typing.

HyperSQL also supports keyword search, making it possible to locate records that contain keyword(s) within any data field (Figure 2). This mechanism provides an alternate way to begin; results are presented in the same hypertext-style format as results from a database search. Keyword search is particularly useful when the information desired is located in unexpected places or embedded in descriptions, notes, or memo fields. Query results are displayed as a list of "headlines" on the results screen (Figure 3). Headlines are single line summaries giving a general identifier for each database record that matched the query criteria. At this point, users may choose to explore one of the headlines (by clicking on it), or may return to the query form to revise the search. (It is always possible to return to the previous screen using the `Back` key provided on the browser). All forms and output screens can also be saved or printed using the browser's facilities.

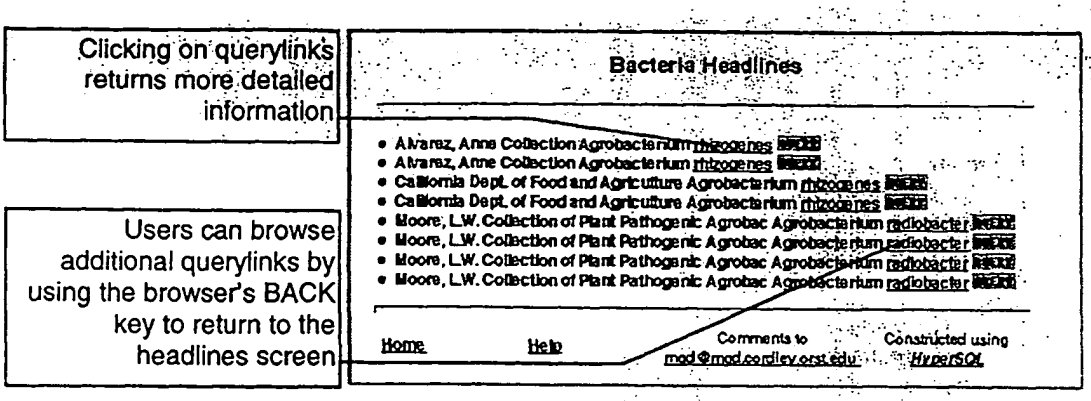


Figure 3. Example of a results screen

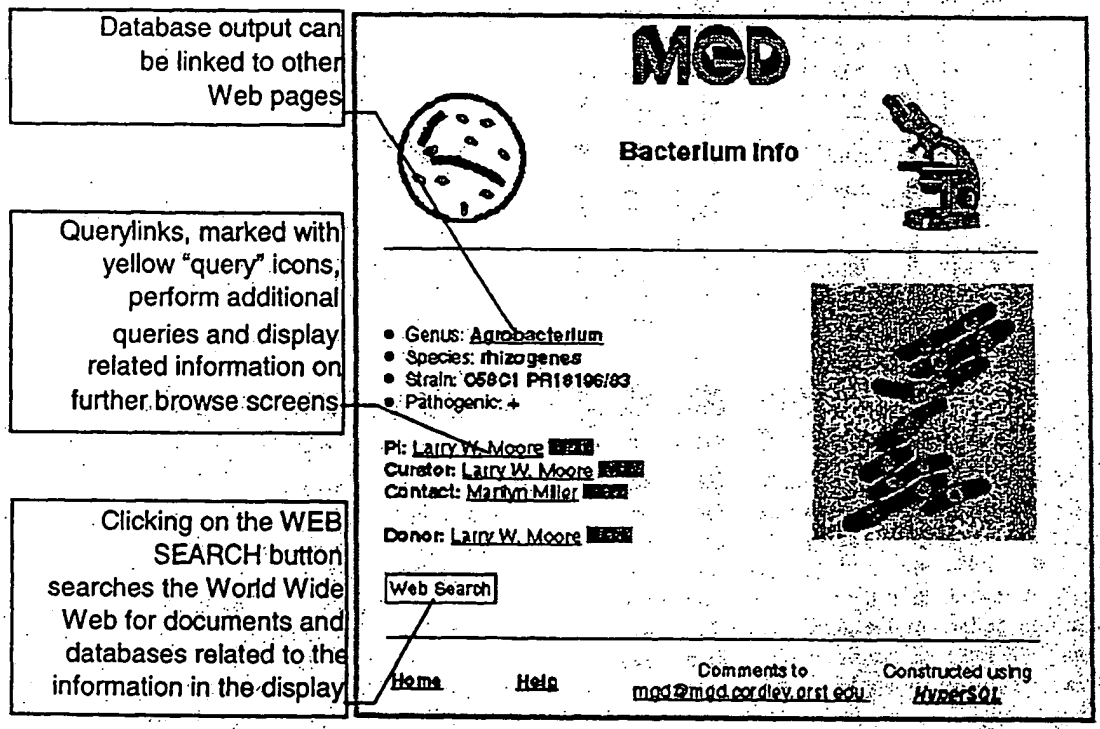


Figure 4. Example of a browse screen

Clicking on a headline brings up a browse screen with more detailed information (Figure 4). A browse screen also provides hyperlinks to access Web documents or other databases and software. In addition, HyperSQL introduces a special style of link called a *querylink* (marked with a yellow "query" icon, as shown in Figure 4), which activates a pre-formulated query to obtain more information from the database. Results from querylinks are displayed on additional browse screens. At the bottom of the browse screen is a Web search button that can be clicked to search the World Wide Web for documents and databases related to the information displayed on the screen.

3. How HyperSQL Query Interfaces are Constructed

This section presents an overview of how HyperSQL query interfaces are constructed. The first subsection discusses how query forms are built, followed by a subsection explaining how SQL is generated from the query forms. The final subsection describes how output is formatted on results and browse screens.

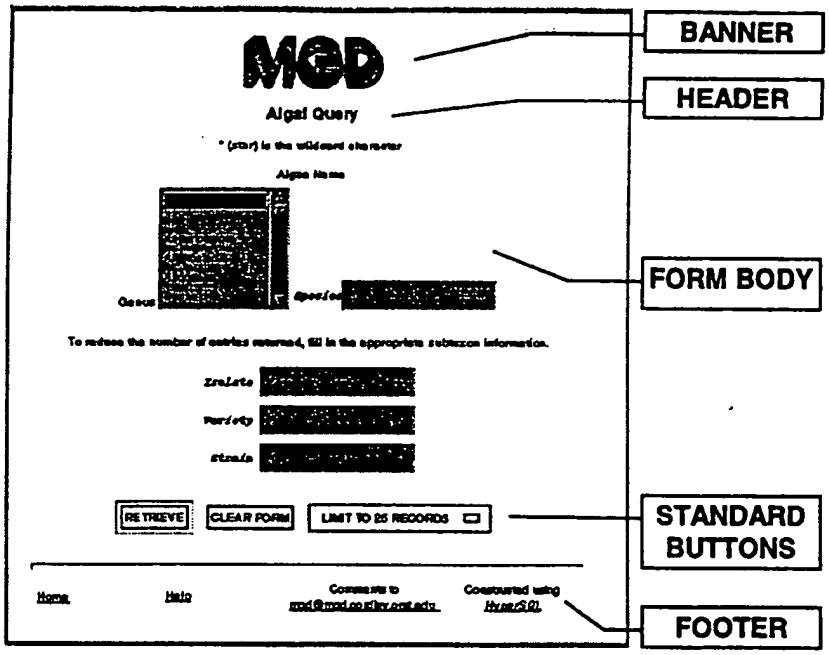


Figure 5. Layout of a sample query form

3.1 Building a Query Form

Query forms are composed of six sections: banner, header, form body, standard buttons, and a footer; Figure 5 shows where these regions appear in the example query form. The layout of the query form is specified in the query file, using "query form" descriptors [NPHM96]. As their names suggest, banner, header, and footer descriptors are used to define what should appear in predefined decoration areas. Since the decoration descriptors accommodate any HTML text, the programmer can include hyperlinks, graphical imagemaps, tables, and formatted text in those areas. If a password is required for the database, HyperSQL automatically prompts for the password before bringing up the query screen.

The *banner* area is located at the top of the screen. It typically displays a logo or database name and is used on all screens in the interface. In contrast, the header (located below the banner) usually varies to reflect the nature of the information on each screen.

The *form body* section specifies what prompts and data entry fields should be presented to the user. HyperSQL's `INPUT` descriptor offers a variety of styles, including radio lights (lists of choices controlled by diamond-shaped buttons), buttons, pulldown menus, and querylists (scrollable lists that are filled with information acquired automatically from the database), as well as simple text entry fields. A row of *standard buttons* automatically appears below the form body. The function of the buttons was described in Section 2.

The *footer* area, located at the bottom of the query form, can be used for a variety of purposes: to display links to the query menu, or related pages; to identify the author and version of the software; or to furnish a button for sending feedback to the developers.

3.2 How SQL is Generated

When the user clicks the `Retrieve` button on the query form, the contents of the form are sent to the HyperSQL interpreter for processing. HyperSQL first establishes a connection to the remote database, using information from the environment section of the query file. (Since connections are short-lived, HyperSQL logs into the host computer and database for every query submission.) Depending on options specified in the "SQL" section of the query file, HyperSQL can either (a) invoke a specified SQL procedure already stored in the database's procedure cache, using a list of values from the query form, or (b) dynamically compose SQL code from the user input and the set of query directives specified in the query file. After transmitting the code to the database, HyperSQL awaits the results.

Errors from the underlying database and the network layer are trapped automatically and displayed prominently at the top of the screen. For debugging, HyperSQL provides a debugging descriptor which can be activated to display all SQL code generated by the HyperSQL interpreter.

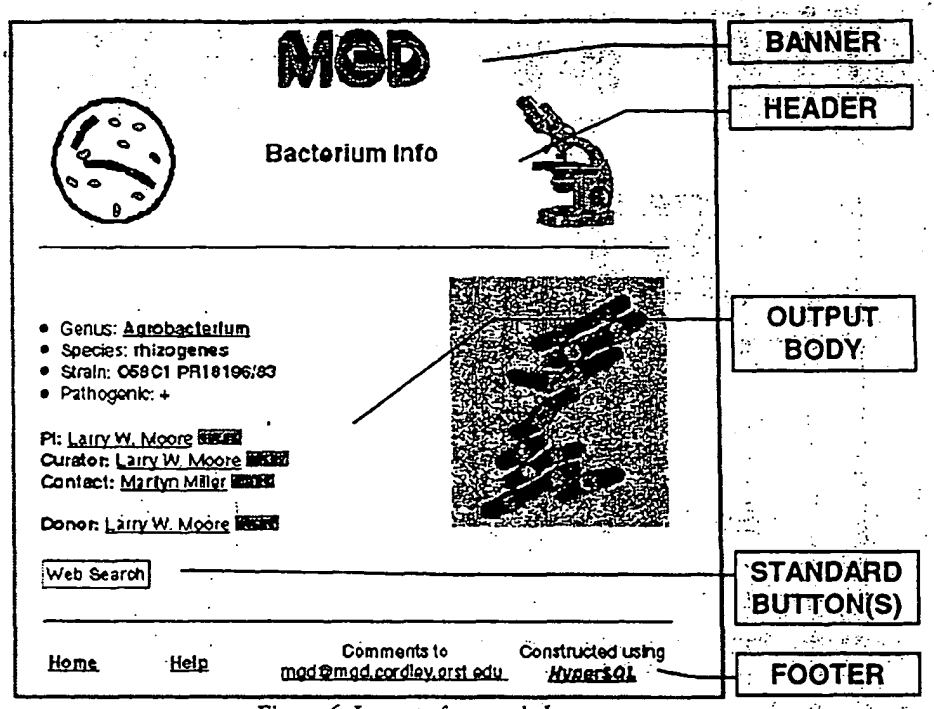


Figure 6. Layout of a sample browse screen

3.3 Formatting Database Output

Query results are formatted as hypertext, using "output" descriptors specified in the query file. There are two types of output screens, **results** and **browse**, controlled by separate sections of the query file. A **results** screen presents a list of single lines, summarizing the results returned from the database or keyword search engine. Data items in the output which cross-reference other information in the database are formatted as querylinks, a special kind of hyperlink designed to retrieve related or more detailed information from the database.

When the user selects a querylink, HyperSQL automatically invokes the associated SQL procedures with selected querylinks, passing the specified data fields as parameters. Since all querylinks on a results screen are statically associated with the same set of SQL procedures, the code is pre-stored in the database's procedure cache to improve response time. Browse screens display the results from following querylinks.

The same output form descriptors are used for both results and browse screens: *banner*, *header*, *form body*, and *footer*. Figure 6 show these regions on an example output screen. Note that output forms are very similar to query forms, except that the *standard buttons* are different and the form body displays output instead of blanks for entering input.

4. Query File Organization

A HyperSQL-based query interface is composed of a set of text files defining the screens. The query menu, which contains only HTML, resides in a file named `qindex.html`. The remainder of the files are called query files and their names have `.hsql` extensions. As a collection they provide complete specifications for a query interface (e.g., Bacterium Query, Fungal Query). Each file contains a combination of HyperSQL descriptors and embedded HTML, organized into five sections (Figure 7). A query file must include exactly one each of the environment, query, and SQL sections, plus one or more output (results and browse) sections. The sections, delimited by BEGIN and END statements, must appear in order:

1. **ENVIRONMENT**: contains information for connecting to the target database and sets the banners for all interface screens.
2. **QUERY**: describes the layout of the query form. It specifies formatting for the form header, input fields on the form body, and the footer.
3. **SQL**: controls how SQL code is generated from information entered on the query form. Alternatively, this section can invoke a SQL procedure stored in the database, or an external script or program.

4. **RESULTS**: describes the layout of query output providing a list of result records. It specifies formatting for a header, query results, and footer.

5. **BROWSE**: describes the layout of screens containing detailed information. It specifies formatting of a header, data fields, and footer.

```
ENVIRONMENT BEGIN
  USEDATABASE mgd LOGIN=MGD PASSWORD=PROMPT;
  BANNER "<center><img src=/HYPERSQL/gif/mgd.gif></center>";
ENVIRONMENT END
QUERY BEGIN
  HEADER "<center> <H2>Bacterium Query</H2></center>";
  INPRINT "<B>Bacterium Name</B><br>";
  INPRINT "  Genus "; INPUT genus   TYPE=TEXTINPUT DEFAULT="Escherichia";
  INPRINT "Species "; INPUT species TYPE=TEXTINPUT DEFAULT="*";
  FOOTER "<hr>Feedback to mgd@mgd.cordley.orst.edu";
QUERY END
SQL BEGIN
  SUB genus   WHERELIST AS upper(Organism.genus)   like upper('$');
  SUB species WHERELIST AS upper(Organism.species) like upper('$');
  FROMLIST    Organism,  Headline_org;
  SELECTLIST  Headline_org.mgd_germplasm_record_num, headline;
  WHERELIST   Organism.mgd_germplasm_record_num = Headline_org.mgd_germplasm_record_num
        AND   Organism.germplasm_type = 'Bacteria';
SQL END
RESULTS BEGIN
 HEADER "<h1><center>Bacterium Headlines</center></h1>";
 OUTPRINT "<br>DATA_FIELD=mgd_germplasm_record_num DATA_FIELD=headline";
 LINK headline TO mgdGetBacterium(mgd_germplasm_record_num) BROWSE_SCREEN=1;
 FOOTER "<hr>Feedback to mgd@mgd.cordley.orst.edu";
RESULTS END
BROWSE 1 BEGIN
 HEADER "<h1><center>Bacterium Screen</center></h1>";
 OUTPRINT SUPPRESS_IF_EMPTY "<li>Genus: <b>DATA_FIELD=Genus</b>";
 OUTPRINT SUPPRESS_IF_EMPTY "<li>Species:<b>DATA_FIELD=Species</b>";
 OUTPRINT "DATA_FIELD=Organism_Role:<b>DATA_FIELD=Researcher_o<br>";
 OUTPRINT "DATA_FIELD=Collection_Role:<b>DATA_FIELD=Researcher_c<br>";
 FOOTER "<hr>Feedback to mgd@mgd.cordley.orst.edu";
BROWSE 1 END
```

Figure 7. Sample HyperSQL query file

5. HyperSQL Implementation

Figure 8 shows the architecture of HyperSQL. The software components of a HyperSQL query interface include the target Sybase database, the HyperSQL gateway, an HTTP (Hypertext Transfer Protocol) daemon, a set of screens (displayed by the browser), and an end-user, assumed to be a scientist or other person who may be unfamiliar with SQL and the target database. The initial version of HyperSQL supports databases using Sybase System 10 [Syb94b], a de facto standard among large-scale scientific databases. Sybase provides multi-database management capabilities and a programmatic interface based on SQL and the Open Client Library [Syb94a], a standard network communication protocol. The Open Client architecture makes it possible for the HyperSQL gateway and target Sybase databases to reside on different computers across the Internet. However, other database vendors have not adopted the interface and we plan to rewrite the communication interface using DBI (Database Interface) [Bun95], a freeware effort under development by a number of third-party database programmers. By taking advantage of DBI we will be able to support other databases, including Oracle, Informix, INGRES, mSQL, Empress, C-ISAM, DB2, Quickbase, and Interbase.

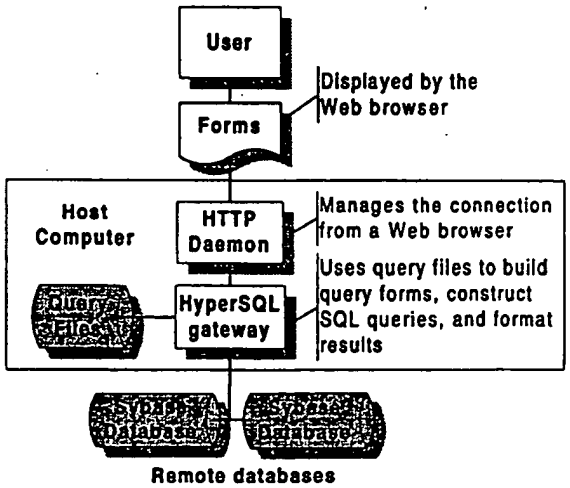


Figure 8. HyperSQL architecture

The core of the HyperSQL software is the *HyperSQL gateway*, composed of an interpreter and a communications back-end. The interpreter carries out the database operations described in the query files, transparently composing SQL queries, establishing RPCs (Remote Procedure Calls) to the remote databases, and formatting database results according to query file specifications (Figure 9). The back-end module contains interface to the Open Client Library. Other communications drivers, such as the Microsoft's Open Database Connectivity and OpenLink's UDBC (Universal Database Connectivity) [Ope95] can be incorporated into the gateway by writing a replacement back-end.

The number of HyperSQL gateways in operation at a given time is limited only by host computer memory, the number of processes running, and the number of users simultaneously logged into the database. Each database transaction (submitting a query form or clicking on a querylink) invokes a new copy of the gateway, which terminates automatically after processing the query results. This style of interaction was chosen over maintaining a continuous login session, because it is not clear in a browser-based interface when a given user-session ends (i.e., interface screens may reside in the browser's cache indefinitely).

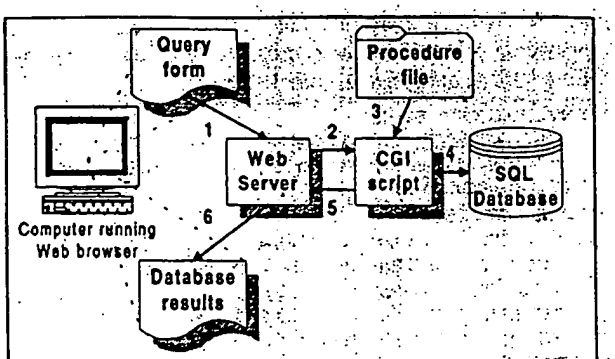
An *HTTP daemon* [NCSA95], available from NCSA and other sources, processes browsers' HTTP (Hypertext Transfer Protocol) requests received on a dedicated machine port (80 hex). The daemon automatically calls the gateway when HyperSQL services are required to build a screen or submit a query. There is a restriction that the daemon and gateway reside on the same machine, because the daemon invokes the gateway as a UNIX child process. The query file must also be accessible to the gateway software.

6. Related Work

Tool developers have constructed a variety of toolkits for constructing browser-based interfaces to remote databases. We have classified the toolkits into three categories, based on how query interfaces are specified: (1) interoperability languages (GSQL, WDB, WebinTool, W3-MSQL), (2) schema-based tools (Zelig, Genera/Web, UMASS Information Navigator), and (3) GUI-based environments (Cold Fusion, dbWeb, Sapphire).

6.1 Interoperability languages

Several toolkit developers use a language approach to interfacing Web browsers to remote databases. Developers construct query files by embedding special directives into HTML (Hypertext Markup Language) text files. Since all of the languages provide minimal features for building forms and generating queries, they differ primarily in number of features, syntax, and programming



1. The Web browser submits the query form to the Web server.
2. The Web server invokes the specialized CGI script commonly written in C or Perl), passing it the contents of the form.
3. The script reads the local procedure file describing how to formulate a query and format the results.
4. The script passes the query to the database and awaits results.
5. The CGI script formats the results into HTML and sends it to the Web server.
6. The Web server sends the results to the browser for display.

Figure 9. How CGI-based query interfaces work

"friendliness". The tools employ a special CGI interpreter to intercept the directives, carry out the corresponding database operations, and format database results into HTML. Figure 9 illustrates how this process works.

The NCSA's (National Center for Supercomputing Applications) GSQL [Ng95] is one of the first Web-to-database interface tools—all the other tools discussed here are considered derivatives of this work. Using GSQL, developers build forms-based query interfaces using a combination of HTML and GSQL commands embedded into "procedure" files. The tool introduced the concept of query directives, used by developers to specify how SQL is to be generated when query forms are submitted. By avoiding vendor-specific SQL extensions and providing a replaceable communications "backend," GSQL can be easily adapted to a variety of SQL databases. GSQL currently supports Oracle, Sybase, and Interbase, but lacks the ability to access remote databases. A restriction of GSQL is the requirement that the software reside on the same machine as the database.

A serious limitation of GSQL is its lack of output formatting. All results are returned as text; no provisions are provided for taking advantage of the browser's capability to display graphics, play sounds, or link output to other Web documents. Furthermore, there are no means of limiting the volume of database output, or linking database fields to related information in the database.

Similar to GSQL, WDB [Ras94] allows developers to specify query interfaces using a set of high-level form definition files, each describing a different view of the database. The WDB script, written in Perl, uses the definition fields to dynamically generate query forms

(where users enter query constraints) and format query results. The definition files permit embedded HTML, as well as calls to Perl functions, for additional formatting. WDB also supplies a utility for extracting the database schema from the database and automatically creating a working template form definition file. Like GSQL, the software must reside on the same machine as the database. WDB currently supports Sybase, Informix, and Mini-SQL, a freely available database which implements a subset of SQL.

WebinTool [Ins95] query interfaces are composed of a web page template and a set of macro directives (prefixed by periods), embedded directly into HTML text files. WebinTool offers a number of enhancements to GSQL: arithmetic operations, variable translation (i.e., uppercase, lowercase, escape, unescape), control directives (.IF .ELSEIF .ELSE, .STOP, .ABORT) and file directives for reading in a file for display (.READFILE). WebinTool's GSQL-style query directives (.SELECT, .FROM, .WHERE, .UNION, .SORT) specify how SQL is to be generated from user input from query forms. References to local and output variables are prefixed with a csh-style '$'; a different prefix, "@", is used to dereference input variables.

Improving on GSQL, WebinTool provides comprehensive support for formatting database output. Template directives (.TPLDEF ... .ENDTPL) specify how output column variables are to be printed (left/right adjusted; maximum length). Data items in the query results can be linked to related information in the database (.ARG LINK). The current version of WebinTool is restricted to machine containing the database software. It supports INGRES, Informix and Mini-SQL, and can support other SQL-based databases by replacing the backend.

Of all the GSQL-style languages, W3-MSQL [Hug95] contains the smallest number of descriptors. Programmers must understand UNIX and C-style concepts; for example, file seek and fetch operations and how to allocate and free handles. Unlike most of the other GSQL-style languages, W3-MSQL provides an "if" construct included primarily to alter output formatting based on the values of data fields returned from the database. W3-MSQL supports Mini-SQL databases and, like the other tools, must be executed from the same machine running the database software.

6.2 Schema-based tools

Each of the tools described in this section is driven by programmer-supplied "schemas," although the meaning of the term is used somewhat differently in each case. Zelig [VH94] uses "schema-based" specifications to guide forms generation. Here "schema" refers to a set of directives embedded into HTML files (coded as comments). The authors take a novel approach to monitoring user behavior in query interfaces. They embed an expert system, written in OPS5, into the interface to accumulate statistics on database access and monitor interface usage patterns. The rule-based module offers the database administrator advice on modifications to the underlying data structures for optimizing access time and storage requirements of the database system.

Another schema-based tool is Genera/Web [Let94]. Using Genera, developers specify query interfaces entirely in the tool's proprietary object-oriented schema language by defining and expressing relationships between "entities," groups of related information in the database. An entity is a collection of fields from one or more tables, views, or other entities, grouped together in a logical set. Fields are permitted to contain links to other databases. From the schema, Genera's compiler generates the HTML code for the query interface and the corresponding SQL procedures to be invoked by the query interface.

Genera, designed for Sybase databases, eliminates the need for developers to code directly in SQL and HTML, but at the expense of having to learn a complex and proprietary schema language. For developers who are already proficient with SQL, Genera introduces an unnecessary level of abstraction, which can be frustrating when the SQL generated from the schema produces undesired results. Although developers can edit the generated SQL stored in the database, Genera automatically overwrites all of the SQL code any time the compiler is invoked—even when simple cosmetic changes are made to the interface.

The third schema-based tool is the UMASS Information Navigator, [Hud95] designed to support easy navigation of relational databases. Hudson's tool generates query forms from a meta-data "schema," which contains information about the organization of the database. After entering search criteria into forms, the tool returns results which in turn can be clicked on to bring up more detailed information in the local database, on other Web pages, or from other databases. The Navigator is ideal for storing an entire Web-based information system. Because it generates Web pages dynamically, the Navigator avoids the "stale link" problem, a major source of frustration of Web users.

A configuration file contains the opening screen information, including the views contained in the meta-database. The Navigator, which supports both Oracle and Basis+, also permits users to use the results of relational queries as starting points for searching using full-text retrieval engines such as WAIS or INQUERY.

6.3 Query Interface Construction Tools

The tools presented in this section are designed to assist programmers in furnishing Web access to ODBC-compliant (Open Database Connectivity) software—which

includes SQL databases and spreadsheets—on Windows 95 and NT platforms. (Similar tools are not yet available on the UNIX platform.)

Cold Fusion [All95] is a CGI-based tool that reads template files to build query interfaces to Windows-based databases. In response to user queries, Cold Fusion creates dynamic HTML pages by mixing HTML tags and Database Markup Language (DBML) from the templates. DBML dictates how queries are formulated and database results are displayed. Using query interfaces created with Cold Fusion, end-users can both update and query databases by interacting with forms displayed on browsers. Cold Fusion provides a number of features targeting financial applications, including special formatting of currency, date, and time fields.

Aspect Software Engineering's dbWeb [Lau95] is a data-driven gateway between ODBC data sources and NT-based web servers implemented as a multithreaded NT service. Forms can be created for both updating and retrieving information from databases using dbWeb's administration tools. The tools generate a combination of HTML and special "tags," (similar to Cold Fusion's DBML) which specify the database operations to be performed by the gateway. At runtime, the gateway intercepts the tags, performs the specified database operations, and wraps the results in HTML for the browser. Database information which contains pointers to related data in the database are formatted as "smartlinks" in the browser, permitting users to navigate and browse the database by clicking links.

Bluestone's Sapphire/Web [Blu95], is a GUI-based programming tool for creating Web-based database interfaces for the three major SQL databases: Sybase, Oracle, and Informix. Sapphire, available for both the X Window System (not yet released) and Windows 95/NT, is similar in operation to Powersoft's PowerBuilder, in that both tools permit the user to build a database interface using direct manipulation. The tool generates C/C++ source code which can be modified or customized by the interface developer.

Sapphire provides no direct support for designing HTML-based forms. Developers can create HTML forms manually using a text editor or by using HTML authoring tools such as HotMetal, WebMagic, or the Internet assistant built into Microsoft Word. Application "objects" such as stored database procedures, dynamic SQL, functions, executables, files, and other objects (i.e., OLE) are dragged and dropped into the "Bind Editor," where HTML elements such as text input fields or drop-down menus are "bound" to arguments; database results are also bound to HTML elements for formatting. Once the interface is defined, the tool generates C and C++, which can be customized manually by programmers, who can introduce conditional processing or change the default method of populating the data. Although the interface does not entirely eliminate programming, it does offer a convenient means of assembling the components of CGI-based database interfaces.

When we started our work, only GSQL and Genera existed. Although GSQL demonstrated how browsers could interoperate with relational databases, it was limited to simple input forms, lacked means for formatting and cross-referencing output to related information, and was unsupported. The drawback to Genera was having to learn its high-level schema language; the tool automatically generates both the interface and SQL. The high degree of automation has a drawback—no provisions are made for queries for which the precise SQL is already known; the programmer must unnecessarily translate SQL into schema language.

Furthermore, most of the subsequently developed tools we have examined do not meet all of our users' requirements for publishing pre-existing, large-scale scientific databases, which are traditionally found on high-end UNIX workstations. In particular, the tools:

- do not connect to remote databases (must be run on the machine running the database) or cannot operate beyond a PC-based local area network;
- do not help users fill out the query form (e.g., displaying a list of allowed values);
- furnish no means of restricting access to sensitive data (e.g., password protection);
- lack means of cross-referencing database results to related data (i.e., querylinks) to help the user visually scan for information of interest.

7. Current Status and Future Work

At the time this paper was prepared, HyperSQL had been successful in building query interfaces in support of three major biological databases:

- Microbial Germplasm Database (MGD) [HNMP95]: searchable information describing microbial germplasm maintained in research collections throughout the U.S.
- Mycological Types Collection (MTC) [PHS96]: searchable descriptions of fungi type specimens.
- Vascular Plants Types Collection (VPTC) [LHP96]: searchable descriptions of vascular plants collections.

These databases have been accessed from more than 14 countries, including Brazil, Canada, Germany, Russia, Sweden, and the US. Feedback from programmers and database end-users have been encouraging. Additional databases are expected to come online using our software in the near future. We are also considering adapting HyperSQL to support Oracle and possibly other databases using DBI (Database Interface) [Bun95].

We have begun development of a next-generation Web tool called QueryDesigner, which will enable non computing professionals to locate, connect, and build personalized query interfaces to remote SQL databases. No knowledge of SQL, HTML, or HyperSQL will be required. Current information on QueryDesigner and HyperSQL, plus a list of supported databases are available at `http://mgd.cordley.orst.edu/hyperSQL`.

References

[All95] Allaire Corporation. *Cold Fusion White Paper*, 1995. [Online]. Available: `http://www.allaire.com/`

[BLCL+94] Tim Berners-Lee, Robert Cailliau, Art Luotonen, Henrik Frystyk Nielsen, and Arthur Secret. The World Wide Web. *Communications of the ACM*, 37(8):76-82, August 1994.

[Blu95] Bluestone Corporation. *Sapphire/Web manual*, 1995. [Online]. Available: `http://www.bluestone.com/products/sapphire/`

[Bun95] Tim Bunce. *The Database Interface DBI: Specifications*, 1995. [Online]. Available: `http://www.hermetica.com/technologia/DBI/`

[FJP90] James C. French, Anita K. Jones, and John L. Pfaltz. Scientific database management (final report). Technical Report TR-90-21, University of Virginia, Department of Computer Science, August 1990.

[HNMP95] Joe Hanus, Mark Newsome, Larry Moore, and Cherri Pancake. "The Microbial Germplasm Database." [Online]. Available: `http://mgd.cordley.orst.edu/cgi-bin/mgd`

[Hud95] Richard L. Hudson. *UMass Information Navigator*. Office of Information Technology, 1995. [Online]. Available: `http://www.crocker.com/~hudson/navigator.html`

[Hug95] David J. Hughes, *W3-MSQL Users Manual*. Hughes Technology Pty, Ltd., 1995. [Online]. Available: `http://hughes.com.au/product/w3-msql/manual-1/w3-msql.htm`

[Ins95] BBSRC Roslin Institute. *WebinTool*, 1995. [Online]. Available: `http:// anita.jax.org/webintool-0.921/docs/user-guide.txt`

[JV85] Matthias Jarke and Yannis Vassiliou. A framework for choosing a database query language. ACM Computing Surveys, 17(3):313-340, May 1995.

[Lau95] Jim Laurel. *dbWeb White Paper*. Aspect Engineering, Inc. August 1995. [Online]. Available: `http://www.aspectse.com/`

[Let94] Stanley Ian Letovsky. *Web/Genera*. John Hopkins Medical Institutes, Baltimore, 1994. [Online]. Available: `http://gdbdoc.gdb.org/letovsky/genera/genera.html`

[...] Peterson, "Vascular Plants Types CollecuonDatabase". [Online]. Available: `http://www.orst.edu/dept/botany/herbarium/vasc_plant.html`

[Lin94] Paul Lindner. GopherSQL, A Gopher Interface to Relational Databases. In *Internet Gopher Conference*, Minneapolis, Minnesota, April 1994.

[NCSA95] NCSA. HTTPd documentation, [Online]. Available: `http:// hoohoo.ncsa.uiuc.edu/docs/Overview.html`

[Ng95] David Ng. *GSQL - a Mosaic-SQL gateway*. NCSA, Champaign, IL, 1995. [Online]. Available: `http://ox.ncsa.uiuc.edu/w/gsql/0gsql.html`

[NPHM96] Mark Newsome, Cherri Pancake, Joe Hanus, and Larry Moore. *HyperSQL User's Guide and Language Reference*, Revised February 1996 [Online]. Available: `http://mgd.cordley.orst.edu/hyperSQL`

[Ope96] OpenLink Software, Inc., 10 Burlington Mall Rd., Suite 265, Burlington, MA 01803, OpenLink ODBC Technical White Paper. [Online]. Available: `http://www.rockhopper.com/openlink`

[PHS96] Sherry Pittam, Joe Hanus, Joey Spatafora, "Mycological Types Collection Database." [Online]. Available: `http://mgd.cordley.orst.edu/hyperSQL/hsql/mct.html`

[Ras94] Bo Frese Rasmussen. WDB: A Web Interface to Sybase. In *Fourth Annual Conference on Astronomical Data Analysis Software and Systems*, Baltimore, September, 1994.

[Sme95] John B. Smelcer. User errors in database query composition. *International Journal of Human-Computer Studies*, 42:353-381, 1995.

[Syb94a] Sybase, Inc. 6475 Christie Avenue, Emeryville, CA 94608. Open Client-Library/C Reference Manual, Revised January 15, 1994.

[Syb94b] Sybase, Inc. 6475 Christie Avenue, Emeryville, CA 94608. Sybase SQL Server Reference Manual, Volume 1, Revised February 1, 1994.

[VH94] Carlos A. Varela and Carolyn C. Hayes. Zelig: A Schema-Based Generation of SoftWWW Database Applications. In *First World Wide Web Conference '94: Mosaic and the Web*, Chicago, September, 1994. NCSA.

[ZI94] Maria Zemankova and Yannis E. Ioannidis. Scientific Databases—State of the Art and Future Directions. In *Proceedings of the Twentieth International Conference on Very Large Databases*, pages 752-753, 1994. Panel.

A Synchronized and Retrievable Video/HTML Lecture System for Industry Employee Training

Herng-Yow Chen[1,2], Jen-Shin Hong[1,2], Yu-Te Wu[1]

Department of Computer Science and Information Engineering[1],
Distance Education Center[2]
National Chi Nan University, Pu-Li, NanTou, Taiwan, 545, R.O.C.
E-mail: hychen@csie.ncnu.edu.tw, jshong@csie.ncnu.edu.tw, ytw@csie.ncnu.edu.tw

Abstract

We present a web-based course system for teaching and training the industry employee. This system is designed to synchronize and retrieve the data related to HTML lecture notes and lecture video. The overall system consists of two frameworks, the Web-based Synchronized Multimedia Lecture (WSML) framework and the XML-Based Synchronized Hypermedia Video Query (XSHVQ) framework. The WSML system synchronizes the presentation of the streaming video lecture, the HTML-based lecture notes and the HTML Navigation Events. It also automates the recording of these three media events based on the Synchronized Multimedia Integration Language (SMIL) specification. To create an online course using this system, teachers first convert their lecture notes into the HTML pages and present the HTML-based lectures in a digital studio. Then the synchronized video clips and the associated HTML-based notes are deposited in our course database for on-demand self-paced learning. The XSHVQ system is designed to efficiently search for the desired video segments based on the synchronization information between video clips and the associated HTML-based notes. To query desired video segments using XSHVQ system, users simply use "keyword" search for the HTML lecture notes and the system can locate the corresponding video segments and present them to users.

1. Introduction

As a result of emerging technologies for multimedia data processing, "WWW-based online courses" is becoming popular for distance education [1-7] and can serve as a major manner for industry employee training. Currently, most online courses provide either unguided HTML pages or course video with blurred images of lecture notes (e.g., RealVideo video streaming systems shown at Fig. 1). Both distance lecture modes have some inevitable downside. Aiming to facilitate the authoring, depositing, and presenting of online course lecture material, we have endeavored to develop various synchronized multimedia lecture systems. In our approach, we use streaming video for the A/V lecture, and use the dynamically loaded HTML pages to present the associate lecture notes since HTML pages can present static text and images in much higher resolution than that presented by a low bit-rate video clip.

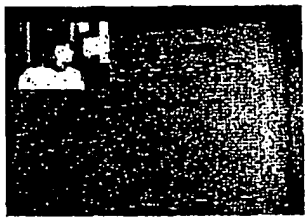


Figure 1. Distorted image of RealVideo at 16Kbps

In this paper, we propose two frameworks, Web-based Synchronized Multimedia Lecture (WSML) framework and XML-Based Synchronized Hypermedia Video Query (XSHVQ) framework, for the purposes of data synchronization and query, respectively. The WSML framework synchronizes the presentation of the streaming video lecture, the HTML-based lecture notes and the HTML Navigation Events. It also automates the recording of these three media events based on the Synchronized Multimedia Integration Language [32] (SMIL) specification. To create an online course using this system, the teachers first convert their lecture notes into HTML pages in advance. They present the HTML-based lectures in a digital studio equipped with camcorder/digitizer for AV lecture recording and digitizing. Then the synchronized video clips and the associated HTML-based notes are deposited in our online course database for on-demand self-paced learning. The implementation of the WSML framework will be introduced in Section 2.

As the amount of online course database increases, efficient search for the desired video segments becomes an important issue. There were various approaches that have been designed to solve the problem based on the "content-based' video query [8-12]. These methods basically apply signal processing or image processing technologies, for example "scene-change detection" or "video segmentation" technologies, to extract the desired video segments. However, most video clips of course lecture in our database contain no significant "scene-change", and thereby current content-based video query algorithms may not be applicable. In this paper, we have developed the XSHVQ framework on top of WSML system to address this issue. We apply the "Metadata" concept [13-16] in our system for structural description and management of the WWW lecture video and HTML lecture notes in our

0-7803-5735-3/99/$10.00 ©1999 IEEE

course lecture database. Let us assume that the synchronization between the video clips and their corresponding HTML pages has been done by the WSML. This synchronization information provides a very convenient and efficient way to index the video segments and can be built into a look-up table to represent the associated correspondences. To query desired video segments using XSHVQ framework, users simply use "keyword" search for the HTML lecture notes and the system can locate the corresponding video segments via look-up table and present them to users. The underlying concept and implementation of the XSML system will be described in Section 3.

2. Web-based Synchronized Multimedia Lecture Framework

Our Web-based, Synchronized AV/HTML Navigation Distance Lecturing Framework consists of three major modules (as shown in Fig. 2): (1) WSML Recorder- for recording the temporal information of the AV lecture and the HTML-based lecture notes navigation process. (2) WSML Event Server- for receiving, depositing, and multicasting/unicasting all WSML events. (3) WSML Browser- for synchronized presentation of the AV lecture and HTML-based lecture notes navigation.

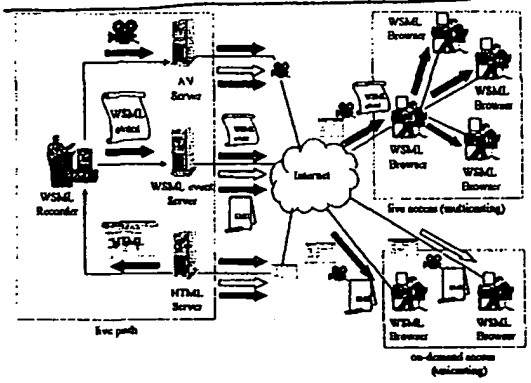


Figure 2. Framework of the proposed WSML

2.1 WSML Recorder

The major function of the WSML Recorder is to record the video together with the associated events during the lecture. The recorder consists of (1) Timer, (2) AV Encoder, (3) Navigation Event Logger (as shown in Fig. 3).

Timer: to initialize the recording process. Each time the teacher starts the recorder, the timer will be initiated and become the time-line of the WSML. At the same time, the timer will automatically activate the AV Encoder and the Navigation Event Logger.

AV Encoder: to encode the AV lecture. It also sends and saves the compressed AV lecture to the AV server for subsequent live broadcasting or lecture on demand.

Navigation Event Logger: to record the temporal information of the AV and the HTML navigation events. The major events occurred during the HTML-based lecture notes navigation might include the mouse event, and the URL event (illustrated in Fig. 4). Those events induced in the teacher site will be sent to the Navigation Events Logger for recording. The mouse events include information regarding mouse drag coordinate, highlighted region, and the scrolling offset. The URL event may occurs at the moments the teacher inputs URL, loads a HTML by pressing a hyperlink, browses backward, or browses forward. The Navigation Event Logging process is exemplified in Fig. 4.

2.2 WSML Event Server

Fig. 5 illustrates the interactions among the WSML Recorder, WSML Event Server, and WSML Browser. The WSML server is responsible for the receiving, depositing, and transmitting of the WSML events. In the live broadcasting lecture mode, The WSML Event Server receives various WSML events from the Navigation Events Logger, and then broadcast to the WSML Browser in the client sites. At the same time, those events will be encoded (by the SMIL gateway) into SMIL 1.0 compliant format to be deposited in the SMIL database for on-demand requests.

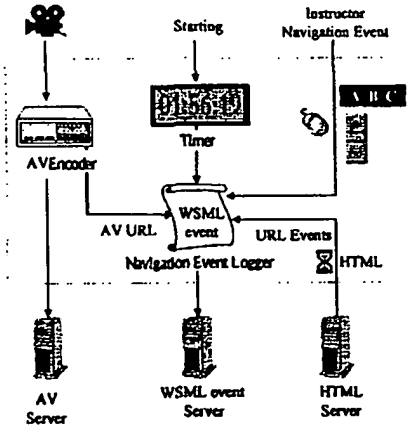


Figure 3. Components of the WSML Recorder

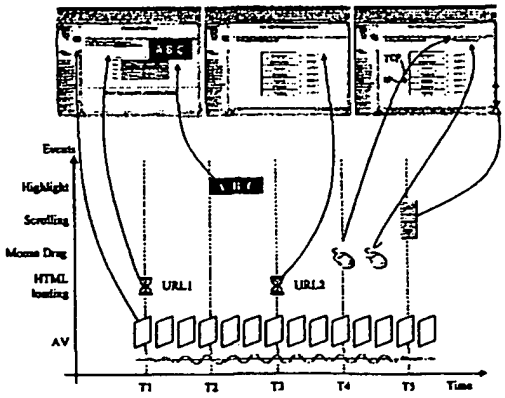


Figure 4. An event logging example.

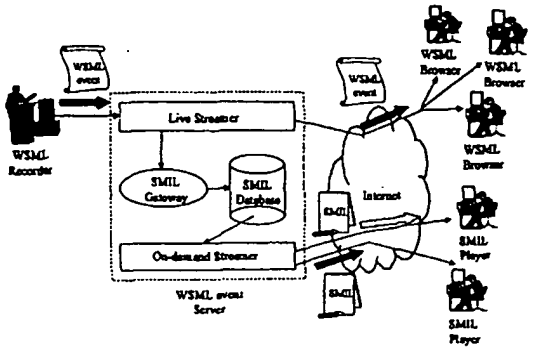


Figure 5. The detailed interactions among the WSML Recorder, WSML Event Server and WSML Browser

2.3 WSML Browser

The WSML Browser is for presenting the AV lecture and replay the corresponding HTML navigation process based on the information broadcast from the WSML Event Server (as shown in Fig. 6). The major components of WSML Browser include WSML Reader, Event Handler, AV player and HTML browser. Take the WSML events illustrated in Fig. 4 as an example, the WSML Browser will present the WSML in the following sequence.

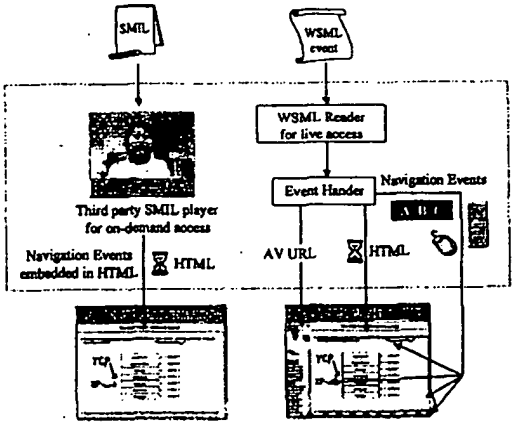


Figure 6. Framework for the WSML Browser

3. The XML-Based Synchronized Hypermedia Video Query (XSHVQ) Framework

To allow users to search efficiently for relevant lecture video segments using keyword-based query, we need to clearly define the relationships between the keyword and video segments. In our framework, we propose the "metadata" concept to describe the synchronization relationships those relevant media. We defined the metadata for describing the temporal relationships between the "Keyword to HTML notes" and "HTML notes to Video Segments".

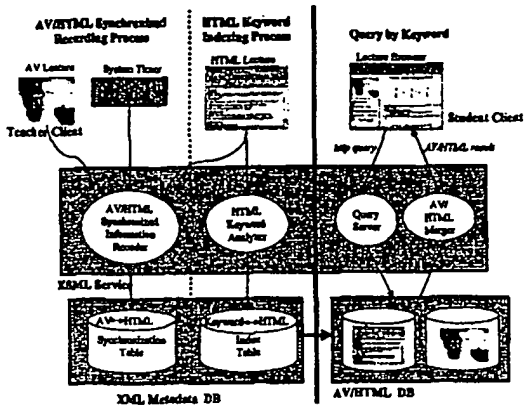


Figure 7. The XSHVQ Framework

On the other hand, as the contents of the online course continue to grow, the metadata for the online course hypermedia will become more complicated. In our design, concerning that the metadata is tree-structured data, we use the XML [17-19], which is a meta language ideally suited for describing the tree-structured data, to define the synchronization meta information. Based on the concept addressed above, we proposed XML-Based Synchronized Hypermedia Video Query (XSHVQ) framework for the video query in our WSML System addressed above. In the proposed XSHVQ framework, we apply XML-based meta information to structurally describe and organize the WWW lecture video and HTML lecture notes in our course lecture database. As shown in Fig. 7, the XSHVQ framework contains four crucial components:

1. **AV/HTML Synchronized Information Recorder:** for logging the synchronization information between the lecture video and the associated HTML-based lecture.
2. **HTML Keyword Analyzer:** for keyword filtering and weighting from the texts in the HTML lecture notes.
3. **Query Server:** for generating and processing the index files of the XML data.
4. **AV/HTML Merger:** for integrating the video segments and the corresponding HTML-based lecture note for presentation.

3.1. AV/HTML Synchronized Information Recorder

The AV/HTML Synchronization Information Recorder is used to automatically log the synchronization information of lecture video segments and associated HTML lecture notes. The synchronization information is formatted according to the XML syntax and deposited into the Synchronization Table (c.f. Fig. 8). During the playback, the system presents to end users the AV segments together with the synchronized HTML pages based on the synchronization information specified inside the Synchronization Table.

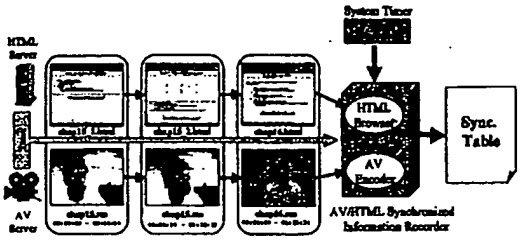


Figure. 8. AV/HTML Synchronized Info. Recorder

The format of the Synchronization Table is compliant to XML syntax (c.f. Table 1.) The tag <HTML id="chap15_1"> indicates the specific ID of the HTML lecture note. The tag <AV_REF idref= "ch15_av" start = "00:00:00" end="00:55:19"/> indicates the starting/ending time of the lecture video segments synchronized with this HTML page.

```
<SYNC_TABLE>
    <HTML id="chap15_1">
        <AV_REF idref="ch15_av"
          start="00:00:00" end=" 00:55:19 "/>
    </HTML>
    <HTML id="chap15_2">
        <AV_REF idref="ch15_av"
        start=" 00:55:19 " end=" 01:30:33 "/>
    </HTML>
    <HTML id="chap16">
        <AV_REF idref="ch16_av"
        start="00:00:00" end="01:55:34"/>
    </HTML>
</SYNC_TABLE>
```

Table 1. The Synchronization Table

3.2. HTML Lecture Notes Keyword Analyzer

For the keyword-based query in the client site, we generate the keyword index [20] automatically for the HTML-based lecture notes. First, all the texts inside the HTML files will be processed by the keyword filter which discards the HTML tags texts and "redundant words" such as "is, that, this..." After the keyword filtering, the remained texts will be processed by the "weighted" index generator (c.f. Fig. 9). The weighting is decided primarily according to the tags encapsulating the word to be indexed. For example, the words inside the HTML tag <TITLE> will possess higher weighting factor that that inside the HTML heading tags (such as <H1>, <H2> and <H6> ...etc.)

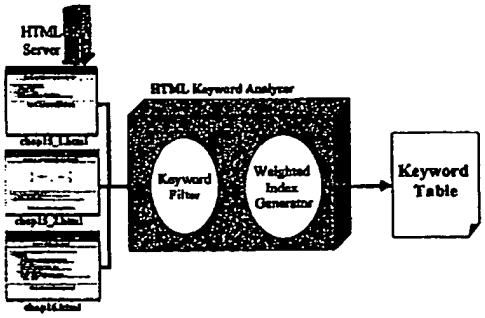


Figure 9. HTML Keyword Analyzer

The keyword index files are formatted based on the XML DTD and deposited in the Keyword Table (Table 2.) As shown in the table 2, <CHAPTER_REF idref="chap15_1" weight=1 /> indicates that page chap15_1 has the keyword "FTP" inside with a weighting factor 1.

```
<KEYWORD_TABLE>
    <KEYWORD id="FTP">
        <CHAPTER_REF idref="chap15_1" weight=1/ >
        <CHAPTER_REF idref="chap16" weight=2/ >
    </KEYWORD>
    <KEYWORD id="IP_address">
        <CHAPTER_REF idref="chap15_2" weight=2/>
        <CHAPTER_REF idref="chap16" weight=4/>
    </KEYWORD>
    <KEYWORD id="virtual_network">
        <CHAPTER_REF idref="chap15_2" weight=3/>
    </KEYWORD>
        ....
</KEYWORD_TABLE>
```

Table 2. The Keyword Table

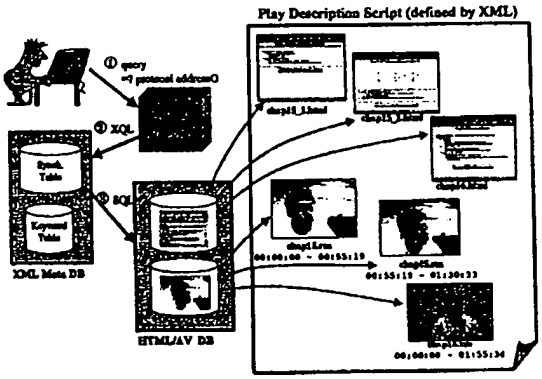


Figure 10. Procedure of the Query Server

3.3. Query Server

The XSHVQ framework applies the XML-based metadata to describe the synchronization information of the multimedia lecture. These metadata are actually "semistructured" and thereby difficult to be managed by traditional relational DBMS [21, 22]. As the lecture videos increase, these XML-formatted metadata could become more complicated and hard to maintain. In our project, we have designed a Query server, as shown in Fig. 10, to generate the index files of XML data and to process the XML data query using the standard XML query language XQL (XML Query Language) [23].

The Sync-Table and the Keyword-Table are the crucial information for students to locate the desired video segments. A typical video query process is exemplified in Fig. 11. As shown in figure, "FTP" is one of the keyword of HTML pages Chap15-1 and Chap16. If a user try to locate all the video segments relevant to "FTP", the system will first use the KEYWORD Table to locate all the HTML lecture note pages contain a keyword "FTP". At the same time, the Query Server can locates the video segments according to the Sync Table which records the

synchronization information between the HTML lecture notes and the lecture video segments. The Video segments and the HTML lecture notes are then merged using the AV/HTML Merger (described below) for the synchronized presentation in the client side.

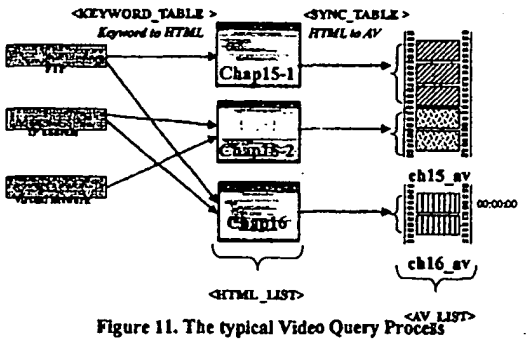


Figure 11. The typical Video Query Process

In our design, the user-specified keyword is transferred to the Query Server through standard http protocol. As shown in Fig. 10, the Query Server then encodes the keyword query into XQL (XML Query Language) compliant format and then transfer to the XML DBMS. Based on the information in the Synchronization Table and Keyword Table, the XML DBMS then encodes the XQL-based query pattern from the Query Server into SQL-compliant format and then transmits to the Multimedia DBMS to extract the information desired. After the HTML and Video extraction, the Query Server will generate an XML-based Play Description Script containing all video segments and HTML URL relevant to the specified Keyword. The Play Description Script will then transferred to the AV/HTML Merger for subsequent processing.

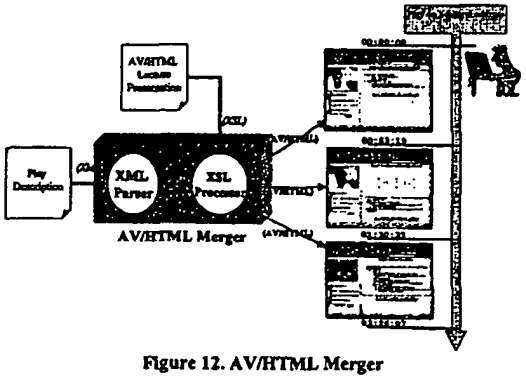


Figure 12. AV/HTML Merger

3.4. AV/HTML Merger

The AV/HTML Merger is responsible for integrating the HTML and AV for synchronous presentation in the user's site. The Merger first parses the XML-based Play Description Script receiving from the Query Server and then fetches the HTML pages and Lecture Video files for presentation style formatting. The Merger uses XSL [24, 25] to format the output into HTML pages. The detailed process of the AV/HTML merging process is illustrated in Fig. 12.

An example of Play Description Script is illustrated in Table 3. The Play Description will sort the video segments presentation according the weighting factor of the indexed keyword assigned in the HTML Keyword Analyzer.

```
<PLAY_DESCRIPTION>
    <QUERY keyword="protocol_address"/>
    <RESULT>
        <HTML id="chap15_1">
            <AV_REF idref="ch15_av"
             start="00:00:00" end=" 00:55:19 "/>
        </HTML>
        <HTML id="chap15_2">
            <AV_REF idref="ch15_av"
             start=" 00:55:19 " end=" 01:30:33 "/>
        </HTML>
        <HTML id="chap16">
            <AV_REF idref="ch16_av"
             start="00:00:00" end="01:55:34"/>
        </HTML>
    </RESULT>
</PLAY_DESCRIPTION>
```

Table 3. Play Description

4 System Implementation

To evaluate the feasibility of the proposed framework, we have built a prototype using RealAudio System [26]. The RealPlayer plug-ins embedded in the Netscape Navigator is used as the AV player in the WSML Browser. The RealServer is used as the AV server and the RVEncoder as the AV Encoder respectively in the WSML Recorder. The synchronization of AV the HTML pages is implemented using the "rmmerge" provided by the RV Encoder which actually embeds the HTML URL events into the AV bit stream.

We use the Java XML Parser [28] and XMJ4J [27] as tools to develop the system. The GAIS [20] is used to generate the index table for the HTML-based lecture notes. The Java Servlet technology [28] are used to generate the query result using the dynamic HTML pages to be sent to the client.

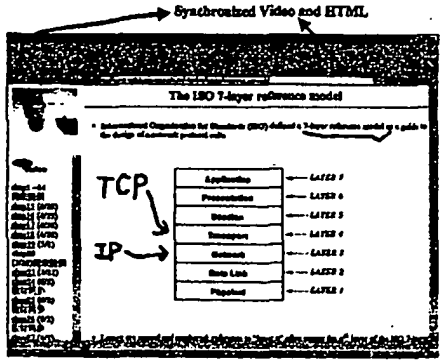


Figure 13. Snapshot of the WSML System

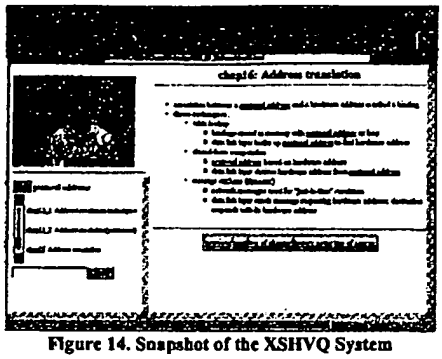


Figure 14. Snapshot of the XSHVQ System

Currently, the proposed XSHVQ system is integrated with our WSML system. A snapshot of both systems, navigation process in WSML system, and typical query results in XSHVQ system, are illustrated in Fig. 13 and Fig. 14, respectively. In this example, a user desired to query all the video segments with content relevant to "protocol address". The user used the keyword "protocol address" to query the XSHVQ system. The XSHVQ system responds the query and displaying all the HTML notes containing the keyword "protocol address". The user is then allowed to click the pages to request the HTML/AV for synchronous presentation.

5. Conclusion

In this paper, we present a synchronized and retrievable Hypermedia course system for industry employee training. The WSML framework has been designed to synchronize the presentation of the streaming video lecture, the HTML lecture notes, and the HTML Navigation Events. The system automates the recording of those media events and significantly reduces the cost for producing multimedia course. Integrated with the WSML framework, the XSHVQ framework provides an efficient and query tool. It allows users to use "keyword" search for desired video segments or HTML lecture notes in our synchronized database. The prototype of this system was implemented and its feasibility was certified.

Reference

[1] Kiran R. Desai, Richard S. Culver, "Multimedia Hypertext on the WWW and its use in Education", FIE' 98 Proceedings

[2] Zhigang Chen, See-Mong Tan, Roy H. Campbell, Yongcheng Li, "Real Time Video and Audio in the World Wide Web", the 4th International World Wide Web Conference, 1995.

[3] Jackey Cheung, Khaled Ben Letaief, and Philip Chan, "Multimedia-Teaching-on-the-Web", proceedings of IEEE Third International Conference on Multi-Media Engineering and Education (MMEE98), Vol. 18, Hongkong, 1998.

[4] Maly K., Abdel-Wahab, H., Overstreet, C.M., Gupta, A,K., Youssef, A., Stoica, E., "Interactive distance learning over intranets", IEEE Internet Computing, pp. 60-71, 1997

[5] Kuang-Chih Lee, Keh-Ning Chang, Shiuh-Sheng Yu, Ing-Chau Chang, Chee-Wen Shia, Wen-Chin Chen, and Jau-Hsiung Huang, "Design and Implementation of Important Applications in a Java-based Multimedia Digital Classroom," IEEE Transactions on Consumer Electronics, vol. 43, no. 3, pp. 264 -270, Aug. 1997.

[6] Graham, C.R.; Trick, T.N., "An innovative approach to asynchronous learning using Mallard: application of Java applets in a freshman course," Frontiers in Education Conference, 1997. 27th Annual Conference. Teaching and Learning in an Era of Change. Proceedings.

[7] Herng-Yow Chen, Chi-Wei Chin, Ging-Yi Chen, Sheng-Wen Shih, and Jen-Shin Hong, "A WWW-based Framework for Language Listening Comprehension Training," published in IEEE International Conference on Multimedia Engineering and Education, July 1998, Hong Kong.

[8] Junichi Takahashi, et al., "Global digital museum: multimedia information access and creation on the Internet", DL '98. Proceedings of the third ACM Conference on Digital libraries , June 23-26, 1998, Pittsburgh, PA.

[9] Aho, S.-F. Chang, K. McKeown, D. Radev, J. Smith, and K. Zaman, "Columbia Digital News Systems: An Environment for Briefing and Search over Multimedia Information", in Internaltional Journal of Digital Library, 1998.

[10] J. R. Smith and S.-F. Chang, "VisualSEEk: A Fully Automated Content-Based Image Query System,", Proc. of ACM Multimedia Conference, Boston, MA, Nov. 1996.

[11] S.-F. Chang, W. Chen, H.J. Meng, H. Sundaram, and D. Zhong, "VideoQ- An Automatic Content-Based Video Search System Using Visual Cues", Proc. of ACM Multimedia Conference, Nov. 1997, Seattle, WA, also Columbia University/CTR Technical Report, CTR-TR #478-97-12.

[12] S. Pack, A. B. Benitez and S.-F. Chang, "Self-Describing Schemes for Interoperable MPEG-7 Content Indexing", Prof. Of IEEE /SPIE Visual Communications and Image , Jan. 1999, San Jose, CA

[13] Rachel Heery," Review of Metadata Formats", Program, Vol. 30, No. 4, October 1996, pp. 345-373

[14] Claude Huc, Thierry Levoir, and Michel Nonon-Latapie, "Metadata: models and conceptual limits" , Proc. Of the Second IEEE Metadata Conference

[15] Stuart Weibel, Jean Miller, Ron Daniel. OCLC/NCSA metadata workshop report. OCLC, March 1995.http:// www.oclc.org:5046 /conferences/metadata/dublin-core-report.html

[16] Metadata, http://www.w3.org/Metadata/

[17] Jon Bosak, "XML, Java, and the future of the Web", web site http://sunsite.unc.edu/pub/sun info/standards/xml/why/xmlapps.htm

[18] XML TECHENOLOGY, http://www.w3.org/ XML/

[19] The SGML/XML Web Page ,http://www.oasis-open.org/cover/,

[20] "GAIS WWW Query Page", web site http:// gais.cs.ccu.edu.tw

[21] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom. "Lore: A Database Management System for Semistructured Data." SIGMOD Record, 26(3):54-66, September 1997.

[22] J. McHugh, J. Widom, S. Abiteboul, Q. Luo, and A. Rajaraman. "Indexing Semistructured Data.", Technical Report, January 1998.

[23] QL'98 - The Query Languages Workshop http://www.w3.org/TandS/QL/QL98/Overview.html

[24] XSL, http://www.w3.org/Style/XSL/

[25] Java XSL API(LotusXSL), http://www.alphaWorks.ibm.com/formula.nsf/xmltechnology/LotusXSL

[26] RealNetworks Inc, http://www.realaudio.comSynchronized Multimedia Integration Language (SMIL) 1.0 Specificatidn, http://www.w3.org/TR/REC-smil

[27] "IBM XML Web Site, Resources - XML Parser in Java.", web site http://www.software.ibm. com/xml/resources/xml-parser.html

[28] JAVA TECHENOLOGY, http://www.javasoft.com

[29] JavaScript, http://www.netscape.com

[30] "Java, JavaScript and plug-in Interaction Using Client-Side LiveConnect", REAZ HOQUE, Technology Evangelist

[31] Java Media Framework (JMF 1.0), http://java.sun.com/products/java-media/jmf/

[32] Synchronized Multimedia Integration Language (SMIL) 1.0 Specification, http://www.w3.org/TR/REC-smil

Organization 1C2600 Bldg./Room PK1
U. S. DEPARTMENT OF COMMERCE
COMMISSIONER FOR PATENTS
P.O. BOX 1450
ALEXANDRIA, VA 22313-1450
IF UNDELIVERABLE RETURN IN TEN DAYS

OFFICIAL BUSINESS

AN EQUAL OP

ARLINGTON JAN 13'05 VA

U.S. OFFICIAL MAIL
U.S. POSTAGE
PENALTY FOR PRIVATE USE $300
05.05
METER